PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-195755
(43)Date of publication of application : 09.07.2003

(51)Int.Cl. G09C 1/00

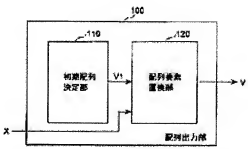
(21)Application number : 2002-299549
(22)Date of filing : 11.10.2002

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD
(72)Inventor : FUDA YUICHI
OMORI MOTOJI
YOKOTA KAORU
TATEBAYASHI MAKOTO

(30)Priority
Priority number : 2001321651 Priority date : 19.10.2001 Priority country : JP

(54) ARRAY OUTPUT DEVICE, ARRAY OUTPUT METHOD, ENCRYPTION DEVICE AND DECRYPTION DEVICE

(57)Abstract:
PROBLEM TO BE SOLVED: To provide an array output device that outputs a well-balanced n- dimensional array based on an integer value such as an output value of a hash function value without using so much memory.
SOLUTION: The array output device is provided with an initial array decision unit 110 that tentatively decides an initial decision array V1 having n1 piece of 1, n2 piece of -1 and (n-n1-2) piece of 0 as its array element, and an array element replacement unit 120 that changes the array element of the initial decision array V1 decided by the initial array decision unit 110 based on an input integer X and that outputs the array V.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2003－195755
（P2003－195755A）
(43)公開日　平成15年７月９日(2003.7.9)

(51)Int.Cl.⁷	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０９Ｃ　1/00	５２０	Ｇ０９Ｃ　1/00	６２０Ｚ　５Ｊ１０４

審査請求　未請求　請求項の数32　ＯＬ　（全 26 頁）

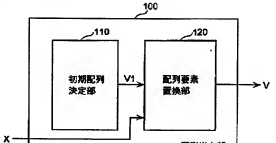
(21)出願番号　特願2002－299549(P2002－299549)
(22)出願日　平成14年10月11日(2002.10.11)
(31)優先権主張番号　特願2001－321651(P2001－321651)
(32)優先日　平成13年10月19日(2001.10.19)
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地
(72)発明者　布田　裕一
大阪府門真市大字門真1006番地　松下電器産業株式会社内
(72)発明者　大森　基司
大阪府門真市大字門真1006番地　松下電器産業株式会社内
(74)代理人　100109210
弁理士　新居　広守

最終頁に続く

(54)【発明の名称】　配列出力装置、配列出力方法、暗号化装置、および復号化装置

(57)【要約】
【課題】　多くのメモリを使用することなく、ハッシュ関数値の出力値などの整数値に基づいて一様にｎ次元の配列を出力する配列出力装置を提供する。
【解決手段】　ｎ１個の配列要素が１であり、ｎ２個の配列要素が－１であり、（ｎ－ｎ１－ｎ２）個の配列要素が０である初期決定配列Ｖ１を暫定的に決定する初期配列決定部１１０と、入力された整数Ｘに基づいて、初期配列決定部１１０が決定した初期決定配列Ｖ１の配列要素を改編し、配列Ｖを出力する配列要素置換部１２０とを備える。



【特許請求の範囲】

【請求項１】　入力された整数に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力装置であって、

初期配列を暫定的に決定する初期配列決定手段と、

前記入力された整数に基づいて、前記初期配列決定手段が決定した前記初期配列の配列要素を改編する改編手段とを備えることを特徴とする配列出力装置。

【請求項２】　前記初期配列決定手段は、Ｋ値の整数の組み合わせからなるｎ次元の配列の１つを初期配列として暫定的に決定し、

前記改編手段は、前記入力された整数に基づいて、前記初期配列決定手段が決定した前記初期配列の配列要素を置換し出力することを特徴とする請求項１記載の配列出力装置。

【請求項３】　前記初期配列決定手段は、Ｋ値のうちの同じ整数がそれぞれ連続した位置に配置されている配列を初期配列として決定することを特徴とする請求項２記載の配列出力装置。

【請求項４】　前記ｎ次元の配列は、ｎ１個の配列要素が整数Ｐ１、ｎ２個の配列要素が整数Ｐ２、・・・、ｎｋ個の配列要素が整数ＰｋであるＫ値の整数の組み合わせからなることを特徴とする請求項３記載の配列出力装置。

【請求項５】　前記改編手段は、

前記入力された整数を所定の整数で除算し剰余を求める除算部と、

前記除算部が求めた剰余に基づいて、前記初期配列の配列要素を置換する置換部とを備えることを特徴とする請求項４に記載の配列出力装置。

【請求項６】　前記除算部は、前記入力された整数を所定の整数で除算したときの商を被除数としてさらに所定の整数で除算することを繰り返し、

前記置換部は、前記除算部が除算毎に求める剰余に基づいて、前記初期配列の配列要素を順次置換することを特徴とする請求項５記載の配列出力装置。

【請求項７】　前記置換部は、前記初期配列の配列要素のうち、前記除算部が除算したときの除数Ｊ番目の配列要素と、前記除算部が除算したときの剰余Ｒ＋１番目の配列要素とを置換し、さらに置換された配列においても前記除算部が繰り返す除算毎に求める除数および剰余に基づいて同様に順次置換することを特徴とする請求項６記載の配列出力装置。

【請求項８】　前記除算部は、除数をｎから２までの各場合において順に除算を行い、

前記置換部は、前記除算部が行う除算毎に前記初期配列を順次置換することを特徴とする請求項７記載の配列出力装置。

【請求項９】　前記ｎ次元の配列は、１，－１，０の３値の組み合わせからなる配列であることを特徴とする請求項８記載の配列出力装置。

【請求項１０】　前記初期配列決定手段は、すべての配列要素がＫ値のうちのいずれかの整数Ｐ３である配列を初期配列として暫定的に決定し、

前記改編手段は、前記初期配列決定手段が決定した前記初期配列における整数Ｐ３の配列要素のうち前記入力された整数に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換えて出力することを特徴とする請求項１記載の配列出力装置。

【請求項１１】　前記ｎ次元の配列は、ｎ１個の配列要素が整数Ｐ１、ｎ２個の配列要素が整数Ｐ２、・・・、ｎｋ個の配列要素が整数ＰｋであるＫ値の整数の組み合わせからなることを特徴とする請求項１０記載の配列出力装置。

【請求項１２】　前記改編手段は、

前記入力された整数を所定の整数で除算し剰余を求める除算部と、

前記初期配列における整数Ｐ３の配列要素のうち前記除算部が求めた剰余に基づく位置の配列要素を整数Ｐ１に置き換える整数配置部とを備えることを特徴とする請求項１１に記載の配列出力装置。

【請求項１３】　前記除算部は、前記入力された整数を所定の整数で除算したときの商を被除数としてさらに所定の整数で除算することを繰り返し、

前記整数配置部は、整数Ｐ１がｎ１個となるまで前記初期配列における整数Ｐ３の配列要素のうち前記除算部が除算毎に求める剰余に基づく位置の配列要素を整数Ｐ１に順次置き換えることを特徴とする請求項１２記載の配列出力装置。

【請求項１４】　前記除算部は、除数をｎから減少させた各場合において順に除算を行い、

前記整数配置部は、前記初期配列における整数Ｐ３の配列要素のうち、前記除算部が除算したときの剰余Ｒ＋１番目の配列要素を整数Ｐ１に順次置き換えることを特徴とする請求項１３記載の配列出力装置。

【請求項１５】　前記改編手段はさらに、前記整数配置部が前記初期配列のｎ１個の配列要素を整数Ｐ１に置き換えた配列における整数Ｐ３の配列要素のうち、さらにｎ２個の配列要素を前記除算部が求める剰余に基づいてＫ値のうちの整数Ｐ２に置き換える第２の整数配置部を備えることを特徴とする請求項１３に記載の配列出力装置。

【請求項１６】　前記ｎ次元の配列は３値の組み合わせからなる配列であり、前記整数Ｐ１、整数Ｐ２、整数Ｐ３は、それぞれ１，－１，０のうちいずれかであることを特徴とする請求項１５記載の配列出力装置。

【請求項１７】　前記入力された整数は複数のビット情報で示されており、

前記改編手段は、前記入力された整数を、所定数のビット情報からなる個々の分割情報に分割する分割部と、

前記初期配列における整数Ｐ３の配列要素のうち、前記分割情報に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換える第３の整数配置部とを備えることを特徴とする請求項１０記載の配列出力装置。

【請求項１８】　前記ｎ次元の配列は、ｎ１個の配列要素が整数Ｐ１、ｎ２個の配列要素が整数Ｐ２、・・・、ｎｋ個の配列要素が整数ＰｋであるＫ値の整数の組み合わせからなることを特徴とする請求項１７記載の配列出力装置。

【請求項１９】　前記第３の整数配置部は、前記初期配列における整数Ｐ３の配列要素のうち、前記個々の分割情報が示す整数に基づく位置の配列要素をＫ値の他の整数Ｐ１に順次置き換えることを特徴とする請求項１８記載の配列出力装置。

【請求項２０】　前記第３の整数配置部は、前記初期配列における整数Ｐ３の配列要素のうちｎ１個の配列要素が整数Ｐ１となるまで、個々の分割情報が示す整数を順に加算した累積値に基づく位置の配列要素を順次整数Ｐ１に置き換えることを特徴とする請求項１９記載の配列出力装置。

【請求項２１】　前記改編手段はさらに、前記第３の整数配置部が前記初期配列のｎ１個の配列要素を整数Ｐ１に置き換えた配列における整数Ｐ３の配列要素のうち、さらにｎ２個の配列要素を前記分割情報に基づいてＫ値のうちの整数Ｐ２に置き換える第４の整数配置部を備えることを特徴とする請求項２０に記載の配列出力装置。

【請求項２２】　前記ｎ次元の配列は３値の組み合わせからなる配列であり、前記整数Ｐ１、整数Ｐ２、整数Ｐ３はそれぞれ、１，－１，０のうちのいずれかであることを特徴とする請求項２１記載の配列出力装置。

【請求項２３】　入力された整数に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力方法であって、
初期配列を暫定的に決定する初期配列決定ステップと、
前記入力された整数に基づいて、前記初期配列決定ステップで決定した前記初期配列の配列要素を改編する改編ステップとを備えることを特徴とする配列出力方法。

【請求項２４】　前記初期配列決定ステップは、Ｋ値の整数の組み合わせからなるｎ次元の配列の１つを初期配列として暫定的に決定し、
前記改編ステップは、前記入力された整数に基づいて、前記初期配列決定ステップで決定した前記初期配列の配列要素を置換し出力することを特徴とする請求項２３記載の配列出力方法。

【請求項２５】　前記改編ステップは、
前記入力された整数を所定の整数で除算し剰余を求める除算ステップと、
前記除算ステップで求めた剰余に基づいて、前記初期配列の配列要素を置換する置換ステップとを備えることを特徴とする請求項２４に記載の配列出力方法。

【請求項２６】　前記初期配列決定ステップは、すべての配列要素がＫ値のうちのいずれかの整数Ｐ３である配列を初期配列として暫定的に決定し、
前記改編ステップは、前記初期配列決定ステップで決定した前記初期配列における整数Ｐ３の配列要素のうち前記入力された整数に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換えて出力することを特徴とする請求項２３記載の配列出力方法。

【請求項２７】　前記改編ステップは、
前記入力された整数を所定の整数で除算し剰余を求める除算ステップと、
前記初期配列における整数Ｐ３の配列要素のうち前記除算ステップで求めた剰余に基づく位置の配列要素を整数Ｐ１に置き換える整数配置ステップとを備えることを特徴とする請求項２６に記載の配列出力方法。

【請求項２８】　入力された整数に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力するための配列出力方法のためのプログラムであって、
初期配列を暫定的に決定する初期配列決定ステップと、
前記入力された整数に基づいて、前記初期配列決定ステップで決定した前記初期配列の配列要素を改編する改編ステップとをコンピュータに実行させることを特徴とするプログラム。

【請求項２９】　前記初期配列決定ステップは、Ｋ値の整数の組み合わせからなるｎ次元の配列の１つを初期配列として暫定的に決定し、
前記改編ステップは、前記入力された整数に基づいて、前記初期配列決定ステップで決定した前記初期配列の配列要素を置換し出力することを特徴とする請求項２８記載のプログラム。

【請求項３０】　前記初期配列決定ステップは、すべての配列要素がＫ値のうちのいずれかの整数Ｐ３である配列を初期配列として暫定的に決定し、
前記改編ステップは、前記初期配列決定ステップで決定した前記初期配列における整数Ｐ３の配列要素のうち前記入力された整数に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換えて出力することを特徴とする請求項２８記載のプログラム。

【請求項３１】　メッセージを暗号化する暗号化装置であって、
メッセージを一方向関数で演算し、その結果を関数値として出力する関数値出力手段と、
初期配列を暫定的に決定する初期配列決定手段、および前記関数値出力手段が出力する関数値に基づいて、前記初期配列決定手段が決定した前記初期配列の要素を改編する改編手段を備え、前記関数値に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力手段と、
前記配列出力手段が出力する配列を基に、暗号文を生成する暗号文生成手段とを備えることを特徴とする暗号化

装置。

【請求項３２】　暗号文を復号化して元のメーセッジを出力する復号化装置であって、

暗号文を復号し、元のメッセージに対応する復号値を出力する復号手段と、

前記復号手段が出力する復号値を一方向関数で演算し、その結果を関数値として出力する第２の関数値出力手段と、

初期配列を暫定的に決定する初期配列決定手段、および前記第２の関数値出力手段が出力する関数値に基づいて、前記初期配列決定手段が決定した前記初期配列の要素を改編する改編手段を備え、前記関数値に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力手段と、

前記配列出力手段が出力する配列を基に、チェック用暗号文を生成する暗号文生成手段と、

前記暗号文と前記チェック用暗号文が一致するか否かを判定し、一致するときに前記復号手段が出力する前記復号値に対して所定の処理を施し元のメッセージを出力する出力手段とを備えることを特徴とする復号化装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、整数値を配列に変換する配列出力装置に関するものであり、特に、情報セキュリティ技術としての暗号技術、誤り訂正技術、およびデジタル署名技術に利用される配列出力装置に関するものである。

【０００２】

【従来の技術】秘密通信方式とは、特定の通信相手以外に通信内容を漏らすことなく通信を行う方式である。またデジタル署名方式とは、通信相手に通信内容の正当性を示したり、通信元がその本人であることを証明したりする通信方式である。この署名方式には公開鍵暗号と呼ばれる暗号方式を用いる。公開鍵暗号は通信相手が多数の時、通信相手ごとに異なる暗号鍵を容易に管理するための方式であり、多数の通信相手と通信を行うのに不可欠な基盤技術である。簡単に説明すると、これは暗号化鍵と復号化鍵が異なり、復号化鍵は秘密にするが、暗号化鍵を公開する方式である。公開鍵暗号については、岡本　龍明、山本　博資、“現代暗号”、産業図書、１９９７（非特許文献１と呼ぶ）が詳しい。

【０００３】この公開鍵暗号の１つとして、ＮＴＲＵ暗号と呼ばれる暗号がある。このＮＴＲＵ暗号は、楕円曲線暗号に比べ暗号化のコードサイズが小さく、家電用機器で使用される非力なＣＰＵでも実装可能であり、将来有望な暗号方式である。

【０００４】このＮＴＲＵ暗号については、Ｊｅｆｆｒｅｙ Ｈｏｆｆｓｔｅｉｎ，Ｊｉｌｌ Ｐｉｐｈｅｒ ａｎｄ Ｊｏｓｅｐｈ Ｈ．Ｓｉｌｖｅｒｍａｎ，“ＮＴＲＵ：Ａ ｒｉｎｇ ｂａｓｅｄ ｐｕｂｌｉｃ ｋｅｙ ｃｒｙｐｔｏｓｙｓｔｅｍ”，Ｌｅｃｔｕｒｅ Ｎｏｔｅｓ ｉｎ Ｃｏｍｐｕｔｅｒ Ｓｃｉｅｎｃｅ，１４２３，ｐｐ．２６７－２８８，Ｓｐｒｉｎｇｅｒ－Ｖｅｒｌａｇ，１９９８（非特許文献２と呼ぶ）に詳しく述べられている。

【０００５】ここで、ＮＴＲＵ暗号について説明する。一般的にすべての多項式ｆ（Ｘ）は、ｆ（Ｘ）＝$f_0+f_1X+f_2X^2+\cdots+f_{N-1}X^{N-1} \bmod (X^N-1)$で表される。

【０００６】以下、多項式ｆ（Ｘ）をｎ次元のベクトルであるベクトル（$f_0,f_1,f_2\cdots,f_{N-1}$）と対応づけて表す。また、このｎ次元のベクトルのうち、ｎ１個が１、ｎ２個が－１、その他の（ｎ－ｎ１－ｎ２）個が０であるベクトルをＬ（ｎ，ｎ１，ｎ２）と表す。

【０００７】このＮＴＲＵ暗号では、秘密にされる復号化鍵（以下、秘密鍵と称する）ｆ（ｖ），Ｆｐ（ｖ）は、次の式で表される。尚、以下ｆ（ｖ），Ｆｐ（ｖ）などの（ｖ）を付した符号は、多項式を表している。秘密鍵ｆ（ｖ）∈多項式の集合Ｌｆ、多項式の集合Ｌｆ＝Ｌ（２６３，５１，５０）

秘密鍵$Fp(v)=f(v)^{-1} \pmod p$すなわち、多項式の集合Ｌｆは、係数$f_0,f_1,f_2\cdots,f_{N-1}$のうち１が５１個、－１が５０個、０が１６２個である多項式の集合であり、秘密鍵ｆはこの多項式の集合Ｌｆに属する多項式である。またｐは整数であり、例えば３である。

【０００８】一方、公開される暗号化鍵（以下、公開鍵と称する）ｈ（ｖ）は、次の式で表される。

公開鍵ｈ（ｖ）＝秘密鍵$f(v)^{-1}\times$多項式$g(v) \pmod q$

多項式ｇ（ｖ）∈多項式の集合Ｌｇ＝Ｌ（２６３，２４，２４）

ここで、公開鍵ｈ（ｖ）はｑ項式である。また、ｑは例えば2^7の整数である。

【０００９】ＮＴＲＵ暗号では、この公開鍵ｈ（ｖ）を用いて次式を基に暗号化される。この暗号化において、メッセージｍｌ（ｖ）の入力に対して、暗号文ｅｌ（ｖ）が出力される。

暗号文$el(v)=p$　多項式$\phi(v)\times$公開鍵$h(v)+ml(v) \pmod q$

多項式$\phi(v)$∈多項式の集合Ｌϕ＝Ｌ（２６３，１６，１６）

ここで、多項式$\phi(v)$は多項式の集合Ｌϕからランダムに選ばれる。

【００１０】一方、暗号文ｅｌ（ｖ）は、上記秘密鍵ｆ（ｖ），Ｆｐ（ｖ）を用いて次の２段階で復号化され、メッセージｍｌ’（ｖ）が得られる。

（１）$a(v)=$秘密鍵$f(v)\times$暗号文$el(v) \pmod q$

（２）$ml'(v)=$秘密鍵$Fp(v)\times a(v)$（m

ｏｄ　ｐ）

【００１１】ところで、暗号解読の方法には、受動的攻撃と能動的攻撃があり、ＲＳＡ暗号、ＥｌＧａｍａｌ暗号、ＮＴＲＵ暗号などの暗号では、受動的攻撃のみを考慮して作られている。受動的攻撃、能動的攻撃、ＲＳＡ暗号、ＥｌＧａｍａｌ暗号については、「非特許文献１」に詳しく述べられている。

【００１２】また最近では、一般の暗号方式を、どのような攻撃に対しても安全性を高くする暗号アルゴリズム改良技術である安全性証明スキーム変換する安全性証明スキーム化技術が提案されている。

【００１３】その安全性証明スキーム化技術に、ＦＯＳＲＴと呼ばれるハッシュ関数を用いる手法がある。

【００１４】ＦＯＳＲＴ及び、ＮＴＲＵ暗号にＦＯＳＲＴを適用する点については、Ｊｅｆｆｒｅｙ　Ｈｏｆｆｓｔｅｉｎ　ａｎｄ　Ｊｏｓｅｐｈ　Ｈ．Ｓｉｌｖｅｒｍａｎ，“Ｐｒｏｔｅｃｔｉｎｇ　ＮＴＲＵ　Ａｇａｉｎｓｔ　Ｃｈｏｓｅｎ　Ｃｉｐｈｅｒｔｅｘｔ　ａｎｄ　Ｒｅａｃｔｉｏｎ　Ａｔｔａｃｋｓ”，ＮＴＲＵＣｒｙｐｔｏｓｙｓｔｅｍｓ　Ｔｅｃｈｎｉｃａｌ　Ｒｅｐｏｒｔ　＃０１６，２０００（非特許文献３と呼ぶ）、ハッシュ関数については、「非特許文献１」に詳しく述べられている。

【００１５】このＦＯＳＲＴの具体的な方法を次に説明する。このＦＯＳＲＴによる暗号化は、次の３段階で行われ、メッセージＭの入力に対して、暗号文Ｅが出力される。

（第１段階）メッセージＭに乱数Ｒｌが連結され、連結されたメッセージＭ‖Ｒｌを得る。

（第２段階）ハッシュ関数を基に、メッセージＭ‖Ｒｌのハッシュ関数値ｈａを得る。

ｈａ＝Ｈ（Ｍ‖Ｒｌ）

（第３段階）メッセージＭ‖Ｒｌとハッシュ関数値ｈａを暗号化のアルゴリズムを基に暗号化し、暗号文Ｅを得る。

Ｅ＝Ｅｎｃ（Ｍ‖Ｒｌ，ｈａ）

【００１６】次に、ＦＯＳＲＴによる復号化を説明する。

（第１段階）暗号文Ｅを復号化して、メッセージＭ‖Ｒｌ’を得る。

Ｄｅｃ（Ｅ）＝Ｍ‖Ｒｌ’

（第２段階）ハッシュ関数を基に、メッセージＭ‖Ｒｌ’のハッシュ関数値ｈａ’を得る。ｈａ’＝Ｈ（Ｍ‖Ｒｌ’）

（第３段階）メッセージＭ‖Ｒｌ’、ハッシュ関数値ｈａ’、および上記暗号化と同一のアルゴリズムを基に暗号化し、暗号文Ｅ’を得る。

Ｅ’＝Ｅｎｃ（Ｍ‖Ｒｌ’，ｈａ’）

（第４段階）暗号文Ｅと暗号文Ｅ’が一致しない場合は出力なし、一致する場合は、メッセージＭ‖Ｒｌ’をメッセージＭ’と乱数Ｒｌ’に分解し、所望のメッセージＭ’を得る。このＦＯＳＲＴをＮＴＲＵ暗号に適用したときの、暗号化と復号化の方法を次に示す。

【００１７】暗号化は３段階にわたって行われ、入力されたメッセージＭ（ｖ）に対して暗号文ｅ（ｖ）を出力する。（第１段階）メッセージＭ（ｖ）にランダムなベクトルＲ（ｖ）を連結して、メッセージｍ（ｖ）を得る。

ｍ（ｖ）＝Ｍ（ｖ）‖Ｒ（ｖ）（第２段階）ハッシュ関数を基に、メッセージｍ（ｖ）のハッシュ関数値Ｈ（ｍ（ｖ））を計算する。（第３段階）暗号文ｅ（ｖ）＝ｐ・Ｈ（ｍ（ｖ））×公開鍵ｈ（ｖ）＋ｍ（ｖ）（ｍｏｄ　ｑ）の式を基に暗号文ｅ（ｖ）を得る。

【００１８】一方、暗号文ｅ（ｖ）の復号化は次の５段階で行われる。（第１段階）ａ（ｖ）＝秘密鍵ｆ（ｖ）×暗号文ｅ（ｖ）（ｍｏｄ　ｑ）を基に多項式ａ（ｖ）を求める。（第２段階）ｍ’（ｖ）＝秘密鍵Ｆｐ（ｖ）×ａ（ｖ）（ｍｏｄ　ｐ）を基にメッセージｍ’（ｖ）を求める。（第３段階）メッセージｍ’（ｖ）のハッシュ関数値Ｈ（ｍ’（ｖ））を計算し、暗号文ｅ’（ｖ）＝ｐ・Ｈ（ｍ’（ｖ））×公開鍵ｈ（ｖ）＋ｍ’（ｖ）（ｍｏｄ　ｑ）を基に、暗号文ｅ’（ｖ）を求める。（第４段階）暗号文ｅ’（ｖ）と暗号文ｅ（ｖ）が一致するかをチェックする。（第５段階）暗号文ｅ’（ｖ）とｅ（ｖ）が一致する場合に、ｍ’（ｖ）＝Ｍ’（ｖ）‖Ｒ’（ｖ）（Ｍ’（ｖ）は復号されたメッセージ、Ｒ’（ｖ）はランダムなベクトル）に分解し、メッセージＭ’（ｖ）を出力する。

【００１９】このようにＦＯＳＲＴをＮＴＲＵ暗号に適用したときの暗号化、および復号化において、ハッシュ関数値Ｈ（ｍ（ｖ））、Ｈ（ｍ’（ｖ））は、例えばＬ（２６３，１６，１６）で表される多項式の集合Ｌφに属する必要がある。

【００２０】この多項式の集合Ｌφは、その係数ｆ０，ｆ１，ｆ２・・・，ｆＮ－１のうち１が１６個、－１が１６個、０が２３１個であるベクトルの集合に対応づけられる。そのため、ハッシュ関数値に対応させて、この１６個の１，１６個の－１，２３１個の０の３値からなるｎ次元配列を求める必要がある。

【００２１】しかし、ＦＯＳＲＴをＮＴＲＵ暗号に適用したときの暗号化、および復号化において、このハッシュ関数値Ｈ（ｍ（ｖ））、Ｈ（ｍ’（ｖ））は整数値となる。そのため、ＮＴＲＵ暗号にＦＯＳＲＴを適用するために、ハッシュ関数値Ｈ（ｍ（ｖ））、Ｈ（ｍ’（ｖ））に基づく、ｎ１個の要素が１、ｎ２個の要素が－１、その他の要素が０であるｎ次元配列を求めなければならない（ここで、ｎ、ｎ１、ｎ２は正の整数である）。

【００２２】ここで、ハッシュ関数値Ｈ（ｍ（ｖ））、Ｈ（ｍ’（ｖ））に基づくｎ次元の配列を求める方法において、以下に示すような条件を満たす必要がある。

（１）同じ入力に対して、いつも同じ出力が対応する。（２）入力と出力の分布が偏らない。（１）は、同じ入力に対して、異なる値を出力しないことである。（２）は、入力に対して、ある出力値だけ、頻繁に現れることのないということである。ＮＴＲＵ暗号にＦＯＳＲＴを適用する場合は、出力のｎ次元配列を送信者、受信者共に、作成可能でないと復号が不可能であるため、（１）を満たさなければ、暗号方式自体が成り立たない。また、（２）を満たさなければ、ハッシュ関数値の出力値に基づいて一様に配列が出力されないため、ハッシュ関数の入力に対する出力の分布の一様性が保持されない。従って、ハッシュ関数の安全性を落とすことになり、ＦＯＳＲＴを適用した時のＮＴＲＵ暗号の安全性が低下することになる。

【００２３】ここで、１個の要素が１、ｎ２個の要素が－１、その他の要素が０であるｎ次元の配列を求める自明な方法について述べる。図１９は、ｎ次元の配列を求める方法を示すフローチャートである。この変換方法は、入力をハッシュ関数値としての整数Ｘと、ｎ１、およびｎ２とし、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元の配列ＶＪを出力する。以下では、配列ＶＪの（左から）ｉ番目の要素をＶＪ［ｉ］（但しｉは１からｎまでの整数）とする。

【００２４】まず、配列ＶＪを全ての要素が０の配列とする（ステップＳ９０１）。次に、カウンタｃ１’のカウント値ｃ１をｌとする（ステップＳ９０２）。次に、ＶＪ［ｃ１］＝１とする（ステップＳ９０３）。次に、カウンタｃ１’のカウント値ｃ１をカウントアップする（ステップＳ９００４）。次に、カウント値ｃ１＞ｎ１であるかを判定し（ステップＳ９０５）、カウント値ｃ１＞ｎ１でない場合（ステップＳ９０５のＮｏ）、再度、ＶＪ［ｃ１］＝１の処理を行う（ステップＳ９０３）。

【００２５】一方、カウント値ｃ１＞ｎ１の場合（ステップＳ９０５のＹｅｓ）、ＶＪ［ｃ１］＝－１とする（ステップＳ９０８）。次に、カウント値ｃ１をカウントアップする（ｃ１←ｃ１＋１）（ステップＳ９０７）。次に、カウント値ｃ１＞ｎ１＋ｎ２であるかを判定し（ステップＳ９０８）、カウント値ｃ１＞ｎ１＋ｎ２でない場合（ステップＳ９０８のＮｏ）、再度、ＶＪ［ｃ１］＝－１の処理を行う（ステップＳ９０６）。

【００２６】一方、カウント値ｃ１＞ｎ１＋ｎ２の場合（ステップＳ９０８のＹｅｓ）、配列ＶＪを出力し、処理を終了する。

【００２７】この方法では、出力される配列ＶＪは、入力される整数Ｘに依らず、はじめのｎ１個の要素が１、その後のｎ２個の要素が－１、その他の要素が０の配列になっている。

【００２８】一方、秘密通信方式として、送信したいメッセージを鍵で暗号化し、同じ鍵を用いて復号する共通鍵暗号方式がある。共通鍵暗号方式の中には、データの置換操作で暗号文を作成する方法がある。例えば、以下のような方法である。

【００２９】この置換方法は、入力を配列ｍ［１］、ｍ［２］、…、ｍ［ｎ］と、鍵Ｋｅ（正の整数）とし、暗号文ｅ［１］，ｅ［２］，…，ｅ［ｎ］を出力する。以下では、予め２次元テーブルＴａｂ［ｊ］［ｉ］（１≦ｊ≦ｎ！，１≦ｉ≦ｎ）を持っているものとする。

【００３０】まず、カウンタｃ’のカウント値ｃを１に設定する。次に、ｅ［Ｔａｂ［Ｋ］［ｃ］］にｍ［ｃ］を代入する。この処理は、カウンタｃ’のカウント値ｃがｎとなるまで行う。そして、カウンタｃ’のカウント値ｃがｎとなると、暗号文ｅを出力して、処理を終了する。このような置換方法を上述したハッシュ関数値に基づいたｎ次元の配列を求める方法に適用することも考えられるが、ｎ＊ｎ！個のテーブルが必要となる。

【００３１】

【非特許文献１】岡本　龍明、山本　博資、“現代暗号”、産業図書、１９９７

【００３２】

【非特許文献２】Ｊｅｆｆｒｅｙ Ｈｏｆｆｓｔｅｉｎ，ＪｉｌｌＰｉｐｈｅｒ ａｎｄ Ｊｏｓｅｐｈ Ｈ．Ｓｉｌｖｅｒｍａｎ，“ＮＴＲＵ：Ａ ｒｉｎｇ ｂａｓｅｄ ｐｕｂｌｉｃ ｋｅｙ ｃｒｙｐｔｏｓｙｓｔｅｍ”，Ｌｅｃｔｕｒｅ Ｎｏｔｅｓ ｉｎ Ｃｏｍｐｕｔｅｒ Ｓｃｉｅｎｃｅ，１４２３，ｐｐ．２６７－２８８，Ｓｐｒｉｎｇｅｒ－Ｖｅｒｌａｇ，１９９８

【００３３】

【非特許文献３】Ｊｅｆｆｒｅｙ Ｈｏｆｆｓｔｅｉｎ ａｎｄ Ｊｏｓｅｐｈ Ｈ．Ｓｉｌｖｅｒｍａｎ，“Ｐｒｏｔｅｃｔｉｎｇ ＮＴＲＵ Ａｇａｉｎｓｔ Ｃｈｏｓｅｎ Ｃｉｐｈｅｒｔｅｘｔ ａｎｄ ＲｅａｃｔｉｏｎＡｔｔａｃｋｓ”，ＮＴＲＵ Ｃｒｙｐｔｏ Ｓｙｓｔｅｍｓ Ｔｅｃｈｎｉｃａｌ Ｒｅｐｏｒｔ ＃０１６，２０００

【００３４】

【発明が解決しようとする課題】しかしながら、上述したｎ次元の配列を求める自明な方法では、出力が１種類に偏っているため、上述した条件（２）（入力と出力の分布が偏らない）を満たさない。この時、ＦＯＳＲＴを適用したことの効果がなくなる上に、受動攻撃に対しても安全性が弱くなってしまう。従って、この方法を用いて、ＮＴＲＵ暗号にＦＯＳＲＴを適用した時のＮＴＲＵ暗号の安全性が低下するという問題がある。

【００３５】また、上述した共通鍵暗号として使用される置換方法を、ｎ次元配列を求める時に適用しても、メモリテーブルを使用するため、多くのメモリを必要とするという問題がある。

【００３６】本発明は、上述した問題点を鑑みて行われ

たもので、多くのメモリを使用することなく、ハッシュ関数値の出力値などの整数値に基づいて一様にｎ次元の配列を出力する配列出力装置を提供することを目的とする。

【００３７】

【課題を解決するための手段】上記目的を達成するために、本発明に係る配列出力装置は、入力された整数に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力装置であって、初期配列を暫定的に決定する初期配列決定手段と、前記入力された整数に基づいて、前記初期配列決定手段が決定した前記初期配列の配列要素を改編する改編手段とを備えることを特徴とする。

【００３８】また、前記初期配列決定手段は、Ｋ値の整数の組み合わせからなるｎ次元の配列の１つを初期配列として暫定的に決定し、前記改編手段は、前記入力された整数に基づいて、前記初期配列決定手段が決定した前記初期配列の配列要素を置換し出力する構成としてもよい。

【００３９】また、前記初期配列決定手段は、すべての配列要素がＫ値のうちのいずれかの整数Ｐ３である配列を初期配列として暫定的に決定し、前記改編手段は、前記初期配列決定手段が決定した前記初期配列における整数Ｐ３の配列要素のうち前記入力された整数に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換えて出力する構成としてもよい。

【００４０】また、前記入力された整数は複数のビット情報で示されており、前記改編手段は、前記入力された整数を、所定数のビット情報からなる個々の分割情報に分割する分割部と、前記初期配列における整数Ｐ３の配列要素のうち、前記分割情報に基づく位置の配列要素をＫ値の他の整数Ｐ１に置き換える第３の整数配置部とを備える構成としてもよい。

【００４１】また、入力された整数に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力方法であって、初期配列を暫定的に決定する初期配列決定ステップと、前記入力された整数に基づいて、前記初期配列決定ステップで決定した前記初期配列の配列要素を改編する改編ステップとを備えることを特徴とする。

【００４２】また、メッセージを暗号化する暗号化装置であって、メッセージを一方向関数で演算し、その結果を関数値として出力する関数値出力手段と、初期配列を暫定的に決定する初期配列決定手段、および前記関数値出力手段が出力する関数値に基づいて、前記初期配列決定手段が決定した前記初期配列の要素を改編する改編手段を備え、前記関数値に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力手段と、前記配列出力手段が出力する配列を基に、暗号文を生成する暗号文生成手段とを備えることを特徴とする。

【００４３】

【発明の実施の形態】（実施の形態１）以下、本発明における第１の実施の形態に係る暗号化装置について図面を用いて説明する。図１（ａ）は本発明における第１の実施の形態に係る暗号化装置の構成を示すブロック図である。

【００４４】暗号化装置１０は、乱数生成部２０と、連結部３０と、ハッシュ関数部４０と、配列出力部１００と、暗号文生成部５０とを備えており、取得したメッセージＭ（ｖ）、および公開鍵ｈ（ｖ）を基に暗号文ｅ（ｖ）を生成し出力する。以下、メッセージＭ（ｖ），公開鍵ｈ（ｖ），暗号文ｅ（ｖ）などの（ｖ）を付した符号は多項式を示している。また、従来例と同じものには同じ符号を付している。

【００４５】この暗号化装置１０を構成する乱数生成部２０と、連結部３０と、ハッシュ関数部４０と、配列出力部１００と、暗号文生成部５０は、それぞれマイクロコンピュータのソフトウェアにより処理を実行しており、ＣＰＵやメモリなどを用いて処理が行われる。

【００４６】この暗号化装置１０は、従来例において説明したように、ＮＴＲＵ暗号に安全性証明スキームのＦＯＳＲＴを適用して、ＮＴＲＵ暗号の安全性をさらに高くした暗号を生成する。乱数生成部２０は、ランダムなベクトルＲ（ｖ）を生成する。

【００４７】連結部３０は、乱数生成部２０が生成したベクトルＲ（ｖ）とメッセージＭ（ｖ）とを連結して、メッセージｍ（ｖ）を生成し、ハッシュ関数部４０と暗号文生成部５０に出力する。

【００４８】ハッシュ関数部４０は、メッセージｍ（ｖ）を一方向関数であるハッシュ関数で演算しハッシュ関数値Ｈ（ｍ）を求め、配列出力部１００に出力する。ここで、ハッシュ関数は一方向関数であり、ハッシュ関数部４０より出力されるハッシュ関数値Ｈ（ｍ）は整数であり、以下、整数Ｘという。

【００４９】配列出力部１００は、ハッシュ関数部４０より出力される整数Ｘに基づいたｎ次元の配列Ｖを生成し暗号文生成部５０に出力する。

【００５０】暗号文生成部５０は、配列出力部１００から出力されるｎ次元の配列Ｖに対応する多項式φ（ｖ）と、連結部３０からのメッセージｍ（ｖ）と、公開鍵ｈ（ｖ）と、暗号文ｅ（ｖ）＝ｐ多項式φ（ｖ）×公開鍵ｈ（ｖ）＋ｍ（ｖ）（ｍｏｄｑ）の式とを基に暗号文ｅ（ｖ）を生成し、外部に出力する。ｐ，ｑは整数であり、ｐは例えば３、ｑは例えば2^7である。

【００５１】一方、図１（ｂ）は本発明における第１の実施の形態に係る復号化装置の構成を示すブロック図である。

【００５２】この復号化装置１５は、復号部２５と、ハッシュ関数部４５と、配列出力部１０５と、暗号文生成

部５５と、判定部６５と、分割部３５とを備えており、入力された暗号文ｅ（ｖ）、秘密鍵ｆ（ｖ）、Ｆｐ（ｖ）、および公開鍵ｈ（ｖ）を基に、暗号化装置１０が暗号化した暗号文ｅ（ｖ）を復号し、元のメッセージＭ’（ｖ）を出力する。

【００５３】この復号化装置１５を構成する復号部２５と、ハッシュ関数部４５と、配列出力部１０５と、暗号文生成部５５と、判定部６５と、分割部３５は、それぞれマイクロコンピュータのソフトウェアにより処理を実行しており、ＣＰＵやメモリなどを用いて処理を行う。

【００５４】復号部２５は、次式を基に入力された暗号文ｅ（ｖ）より元のメッセージに対応する復号値であるメッセージｍ’（ｖ）を得る。

ａ（ｖ）＝秘密鍵ｆ（ｖ）×暗号文ｅ（ｖ）（ｍｏｄ　ｑ）

ｍ’（ｖ）＝秘密鍵Ｆｐ（ｖ）×ａ（ｖ）（ｍｏｄ　ｐ）

【００５５】ハッシュ関数部４５は、メッセージｍ’（ｖ）のハッシュ関数値Ｈ（ｍ’）を演算し、配列出力部１０５に出力する。ハッシュ関数部４５より出力されるハッシュ関数値Ｈ（ｍ’）は整数であり、以下、整数Ｘ’という。

【００５６】配列出力部１０５は、ハッシュ関数部４５より出力される整数Ｘ’に基づいてｎ次元の配列Ｖ’を生成し暗号文生成部５５に出力する。

【００５７】暗号文生成部５５は、配列出力部１０５から出力されるｎ次元の配列Ｖ’に対応する多項式φ’（ｖ）と、復号部２５からのメッセージｍ’（ｖ）と、公開鍵ｈ（ｖ）と、暗号文ｅ’（ｖ）＝ｐ多項式φ’（ｖ）×公開鍵ｈ（ｖ）＋ｍ’（ｖ）（ｍｏｄ　ｑ）の式とを基に暗号文ｅ’（ｖ）を生成し、判定部６５に出力する。

【００５８】判定部６５は、暗号文ｅ（ｖ）と、暗号文ｅ’（ｖ）とを入力し、両者が一致するかを判定し、一致したと判定するとメッセージｍ’（ｖ）を分割部３５に出力する。

【００５９】分割部３５は、判定部６５からのメッセージｍ’（ｖ）をメッセージＭ’（ｖ）とランダムなベクトルＲ’（ｖ）に分解し、元のメッセージＭ’（ｖ）を出力する。

【００６０】次に、暗号化装置１０における配列出力部１００について図面を用いて説明する。尚、復号化装置１５における配列出力部１０５は、配列出力部１００と同様の構成であり、同様の動作をするため、説明を省略する。

【００６１】図２は、配列出力部１００の構成を示すブロック図である。この配列出力部１００は、整数Ｘを入力とし、Ｌ（ｎ，ｎ１，ｎ２）に属する配列Ｖを出力する。ここで、Ｌ（ｎ，ｎ１，ｎ２）は、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元配列全体であり、このｎ、ｎ１、ｎ２は配列出力部１００に予め設定されているものとする。配列出力部１００は、初期配列決定部１１０と配列要素置換部１２０からなる。

【００６２】初期配列決定部１１０は配列Ｖの初期決定を行うものであり、以下のような初期決定配列Ｖ１を出力する。

（１）Ｖ１［ｉ］＝１（１≦ｉ≦ｎ１）

（２）Ｖ１［ｉ］＝－１（ｎ１＋１≦ｉ≦ｎ１＋ｎ２）

（３）Ｖ１［ｉ］＝０（ｎ１＋ｎ２＋１≦ｉ≦ｎ）

【００６３】ここで、Ｖ１［ｉ］を初期決定配列Ｖ１の（左から）ｉ番目（ｉはｌからｎまでの整数）の配列要素（要素）とする。

【００６４】図３は、配列出力部１００による各段階での配列Ｖの配列状態を示す図である。図３では、例えばＬ（８，３，２）の場合（ｎ＝８，ｎ１＝３，ｎ２＝２）における配列Ｖの各配列状態を示している。図３において、１番上の配列状態は、初期配列決定部１１０により決定された初期決定配列Ｖ１を示している。

【００６５】配列要素置換部１２０は、初期配列決定部１１０より出力された初期決定配列Ｖ１と、整数Ｘを入力とし、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元の配列Ｖを出力する。すなわち、初期配列決定部１１０が決定した初期決定配列Ｖ１の配列要素を改編する。

【００６６】以下、配列要素置換部１２０の行う処理を説明する。図４は、配列要素置換部１２０の行う処理を示すフローチャートである。以下では、カウンタｃ’の値をカウント値ｃとする。

【００６７】まず、配列要素置換部１２０は、整数Ｘを変数Ｖ’に、初期決定配列Ｖ１を配列Ｖ’に代入する。即ち、全てのｉ（ｉはｌからｎまでの整数）に対して、Ｖ１［ｉ］をＶ’［ｉ］に代入する（ステップＳ１０１）。

【００６８】次に、配列要素置換部１２０は、カウンタｃ’のカウント値ｃをｎに設定する（ステップＳ１０２）。

【００６９】次に、配列要素置換部１２０は、変数Ｙをカウント値ｃで割った商をＳ、余り（剰余）をＲとする（ステップＳ１０３）。

【００７０】次に、配列要素置換部１２０は、ｔｍｐ←Ｖ［ｃ］とする（ステップＳ１０４）。すなわち、レジスタｔｍｐに配列Ｖのｃ番目の要素Ｖ［ｃ］を代入する。

【００７１】次に、配列要素置換部１２０は、Ｖ［ｃ］←Ｖ［Ｒ＋１］とする（ステップＳ１０５）。すなわち、配列Ｖの（Ｒ＋１）番目の要素を配列Ｖのｃ番目の要素に代入する。

【００７２】次に、配列要素置換部１２０は、Ｖ［Ｒ＋１］←ｔｍｐとする（ステップＳ１０６）。すなわち、レジスタｔｍｐの内容を配列Ｖの（Ｒ＋１）番目の要素

に代入する。

【００７３】このｔｍｐ＝Ｖ［ｃ］、Ｖ［ｃ］＝Ｖ［Ｒ＋１］、Ｖ［Ｒ＋１］＝ｔｍｐの処理（ステップＳ１０４からステップＳ１０６）により、配列Ｖのｃ番目の要素と（Ｒ＋１）番目の要素が入れ替わる。

【００７４】次に、配列要素置換部１２０は、変数Ｙに商Ｓを代入する（ステップＳ１０７）。次に、配列要素置換部１２０は、カウント値ｃが２であるか否かを判定し（ステップＳ１０８）、カウント値ｃが２であると判定した場合（ステップＳ１０８のＹｅｓ）、配列要素置換部１２０は、配列Ｖを出力して（ステップＳ１１０）、処理を終了する。

【００７５】一方、配列要素置換部１２０は、カウント値ｃが２でないと判定した場合（ステップＳ１０８のＮｏ）、カウント値ｃをカウントダウン（ｃ←ｃ－１）して（ステップＳ１０９）、再度、変数Ｙをカウント値ｃで割った商をＳ、余りをＲとする処理（ステップＳ１０３）に移る。

【００７６】このように、配列要素置換部１２０は、カウンタｃ’のカウント値ｃがｎから２の各場合において、商Ｓ、余りＲを求める処理（ステップＳ１０３）からカウント値ｃをカウントダウンする処理（ステップＳ１０９）までの各ステップの処理を繰り返す。これにより、配列Ｖの各要素が入れ替わり、カウント値ｃ＝２での処理が終了すると、配列要素置換部１２０は配列Ｖを出力する。

【００７７】ここで、配列出力部１００全体の動作を説明する。配列出力部１００は、まず初期配列決定部１１０が、はじめのｎ１個の要素が１、続くｎ２個の要素が－１、その他の要素が０であるｎ次元配列の初期決定配列Ｖ１を決定して、配列要素置換部１２０に出力する。

【００７８】次に配列要素置換部１２０は、初期配列決定部１１０から出力された初期決定配列Ｖ１と、配列出力部１００に入力された整数Ｘを取得し、初期決定配列Ｖ１の各要素を、整数Ｘを基に入れ替えて、ｎ１個の要素が１、ｎ２個の要素が－１、その他の要素が０であるｎ次元の配列Ｖを出力する。

【００７９】この配列出力部１００の動作を、具体例を用いて説明する。この配列出力部１００の動作を説明する際、例えば実際に入力される整数Ｘ＝３９３５６、ｎ＝８、ｎ１＝３，ｎ２＝２を用いて説明する。

【００８０】まず、初期配列決定部１１０は、ｎ＝８、ｎ１＝３，ｎ２＝２を満たす初期決定配列Ｖ１を決定し、配列要素置換部１２０に出力する（図３参照）。そして、配列要素置換部１２０は、入力された整数Ｘを変数Ｙに代入し、変数Ｙ＝３９３５６とし、配列Ｖに初期決定配列Ｖ１を代入する（ステップＳ１０１）。

【００８１】次に、配列要素置換部１２０は、カウント値ｃ＝８とする（ステップＳ１０２）。次に、配列要素置換部１２０は、商Ｓと余りＲを演算し、商Ｓ＝４９１９、余りＲ＝４とする（ステップＳ１０３）。

【００８２】次に、配列要素置換部１２０は、配列Ｖの（Ｒ＋１）番目、すなわち５番目の要素である「－１」と８番目の要素である「０」を入れ替え（ステップＳ１０４～ステップＳ１０５）、図３の配列Ｖ２に示す配列状態とする。次に、配列要素置換部１２０は、変数Ｙに４９１９を代入する（ステップＳ１０７）

【００８３】そして、カウント値ｃ＝８でありカウント値ｃ＝２でないため（ステップＳ１０８のＮｏ）、配列要素置換部１２０は、カウント値ｃをカウントダウンしてカウント値ｃ＝７とする（ステップＳ１０９）。

【００８４】そして、配列要素置換部１２０は、再度、変数Ｙ＝４９１９、カウント値ｃ＝７より、商Ｓおよび余りＲを演算し、商Ｓ＝７０２、余りＲ＝５とする（ステップＳ１０３）。

【００８５】次に、配列要素置換部１２０は、配列Ｖの左から６番目の要素である「０」と７番目の要素である「０」を入れ替え（ステップＳ１０４～ステップＳ１０６）、図３の配列Ｖ３に示す配列状態とする。

【００８６】このように、配列要素置換部１２０は、カウント値ｃがｎから２であるそれぞれの場合において、変数Ｙをカウント値ｃで割ったときの商Ｓと余りＲに基づいて、配列Ｖの要素の入れ替えを行い（ステップＳ１０３→ステップＳ１０９の処理の繰り返し）、初期決定配列Ｖ１の各要素の入れ替え処理を行う。そして、カウント値ｃ＝２である場合の上記処理が終了すると、配列要素置換部１２０は配列Ｖを出力する。

【００８７】ここで、配列出力部１００から出力される配列Ｖが入力される整数Ｘに基づいて一様に分布していることを説明する。配列出力部１００は、０≦Ｘ≦（ｎ！－１）（但しｎ！はｎの階乗）を示す。具体的には、ｎ！＝ｎ×（ｎ－１）×…×２×１である。）を満たす整数Ｘに基づいて一様に配列Ｌ（ｎ，ｎ１，ｎ２）を出力している。以下では、Ｘを０≦Ｘ≦（ｎ！－１）を満たすＸに限定して、説明を行う。

【００８８】配列要素置換部１２０では初期決定配列Ｖ１の要素の入れ替えを行っている。以下では、配列要素置換部１２０が行う初期決定配列Ｖ１の要素の入れ替えについて具体的に見ていく。また以降の説明を分かり易くするために、上述の配列要素置換部１２０の構成において、カウント値ｃがｉ（ｉ＝ｎ，ｎ－１，ｎ－２，…，３，２）の時、Ｒ←Ｙｍｏｄ　ｃ（ステップＳ１０３）の処理による余りをＲ＿ｉとする。

【００８９】全体の説明の流れは、以下の通りである。

（１）入れ替え処理が重複しないことについて説明する。

（２）（１）の結果を利用して、整数Ｘと配列要素置換部１２０における処理内容が１対１に対応していることを説明する。

（３）同じ配列を出力する整数Ｘが何種類あるかを計算

する。
この（３）の計算において配列Ｖに依らず、整数Ｘの種類の数が同じであることを示すことが可能であるため、入力に対して出力が一様に分布することがいえる。
【００９０】まず（１）の「入れ替え処理が重複しないことについて」について説明する。以下では、まず、同じ値に対する入れ替えが複数回起こらない、即ち、入れ替え処理の終わった値は、その処理以降の入れ替えの対象になら ないことについて説明する。
【００９１】・カウンタｃ’のカウント値ｃがｎの時
ステップＳ１０４、ステップＳ１０５、ステップＳ１０６の動作（以下、「カウント値ｃがｎの場合の入れ替え処理」と呼ぶ）が意味するところは、初期決定配列Ｖ１のｎ番目の要素Ｖ１［Ｒ＿ｎ＋１］を入れ替えていることである。ここで、Ｒ＿ｎは、変数Ｙをｎで割った時の余りであるので、０からｎ－１までの値を取りうる。
【００９２】・カウント値ｃがｎ－１の時
この時は、カウント値ｃがｎである時の入れ替え処理を終えた状態における配列Ｖのｎ－１番目の要素と、（Ｒ＿（ｎ－１）＋１）番目の要素を入れ替えている。ここで、Ｒ＿（ｎ－１）は、変数Ｙをカウンタの値（ｎ－１）で割った時の余りであるので、０からｎ－２までの値を取りうる。ゆえに、カウント値ｃがｎ－１である時の入れ替え処理を終えた状態の配列Ｖの（ｎ－１）番目の要素は、カウント値ｃがｎである時の入れ替え処理を終えた時点の配列Ｖの（Ｒ＿（ｎ－１）＋１）番目の要素になっている。また、カウント値ｃがｎである時の入れ替え処理を終えた状態の配列Ｖのｎ番目の要素は、入れ替え対象となっていないことに注意する。
【００９３】・カウント値ｃがｎ－２以下の時
カウント値ｃがｎ－２以下の場合について考える。この場合、カウント値ｃがｎ－１の時と同様である。カウント値ｃがｉの時の入れ替え処理を終えた状態の配列Ｖのｉ番目の要素は、カウント値ｃが（ｉ＋１）の時の入れ替え処理を終えた状態の配列Ｖの（Ｒ＿ｉ＋１）番目の要素になっている。ここで、カウント値ｃが（ｉ＋１）の時の入れ替え処理は、カウント値ｃがｉの時の入れ替え処理の前に行われることを注意しておく。またこれ以降、配列Ｖのｉ番目の要素は、他の値と入れ替えられることはない。なぜならばｊ≦ｉ－１とすると、Ｒ＿ｊは変数Ｙをｊで割った時の余りであるから、Ｒ＿ｊ≦ｊ－１を満たす。よって、（Ｒ＿ｊ＋１）≦ｉ－１＜ｉとなるからである。
【００９４】配列要素置換部１２０の処理が終了した後、すなわち、カウント値ｃがｎから２までの全ての処理を終えた状態での配列Ｖの各要素は、初期決定配列Ｖ１の各要素を入れ替えたものになっている。
【００９５】次に、配列要素置換部１２０の処理の一意性について説明する。以下では、上記の結果（入れ替えが重複しない）を利用して、整数Ｘと配列要素置換部１２０における処理内容が１対１に対応していることを説明していく。
【００９６】・整数Ｘと数列の対応について
整数Ｘを０≦Ｘ≦（ｎ！－１）を満たす整数とする。この時、カウンタｃ’のカウント値ｃがｃの時のステップＳ１０３における変数Ｙの値をＹ＿ｃ、商Ｓの値をＳ＿ｃ、余りＲの値をＲ＿ｃとする（Ｒ＿ｃの値については先に定義したものと同じである）。この時、ステップＳ１０３で行われていることを式で表現すると、Ｙ＿ｃ＝ｃ×Ｓ＿ｃ＋Ｒ＿ｃとなる。但し、０≦Ｒ＿ｃ≦（ｃ－１）である。またステップＳ１０７で行われていることを式で表現すると、Ｙ＿（ｃ－１）＝Ｓ＿ｃとなる。
【００９７】これを、ｃ＝ｎ、（ｎ－１）、…、２の場合に当てはめて整理すると、Ｘ＝Ｙ＿ｎであることに注意して、
Ｘ＝　（ｎ×（ｎ－１）×…×３）×Ｒ＿２＋（ｎ×（ｎ－１））×Ｒ＿（ｎ－２）＋…＋（ｎ×（ｎ－１））×Ｒ＿（ｎ－２）＋ｎ×Ｒ＿（ｎ－１）＋Ｒ＿ｎ
となる。
【００９８】ここで、Ｒ＿ｉは、０≦Ｒ＿ｉ≦ｉ－１を満たす整数である。整数Ｘから、Ｒ＿２，Ｒ＿３，…，Ｒ＿（ｎ－２），Ｒ＿（ｎ－１），Ｒ＿ｎが一意的に決まるのは明らかである。また逆も明らかである。よって、０≦Ｘ≦（ｎ！－１）なる整数Ｘと、Ｒ＿２，Ｒ＿３，…，Ｒ＿（ｎ－２），Ｒ＿（ｎ－１），Ｒ＿ｎは１対１に対応していることが分かる。
【００９９】・数列と入れ替えの対応について
先に述べたように、本実施の形態１においては、初期決定配列Ｖ１を、整数Ｘに従って各要素を入れ替えて配列Ｖを出力するわけである。先にも述べたように、その入れ替えは、数列Ｒ＿ｎ，Ｒ＿（ｎ－１），Ｒ＿（ｎ－２），…，Ｒ＿３，Ｒ＿２により定まる。なぜならば、カウント値ｃがｎである時の入れ替え処理終了時点での配列Ｖのｎ番目の要素は、初期決定配列Ｖ１の（Ｒ＿ｎ＋１）番目の要素である。以降は、配列Ｖのｎ番目の要素は入れ替わらない。同様にして、カウント値ｃがｉの時の入れ替え処理終了時点での配列Ｖのｎ番目の要素は、カウント値ｃが（ｉ＋１）の時の入れ替え処理終了時点での配列Ｖの（Ｒ＿ｉ＋１）番目の要素である。以降は、配列Ｖのｉ番目の要素は入れ替わらない。即ち、Ｖのｎ番目から ｉ番目の要素までは変わらない。またカウント値ｃが（ｉ＋１）時の入れ替え処理終了時点での配列Ｖは、数列Ｒ＿ｎ、Ｒ＿（ｎ－１）、…、Ｒ＿（ｉ＋１）により定まっているからである。
【０１００】また、初期決定配列Ｖ１のｎ個の要素の入れ替え方は、「ｎ個の元の置換」と考えられる。任意の「ｎ個の元の置換」に対して、その置換は上に説明した配列要素置換部１２０の処理内容により実現できる。以下にその説明を行う。置換の表現として、入力であるｎ

個の順序付要素Ｖ［１］，Ｖ［２］，…，Ｖ［ｎ］を置換した結果である順序列を用いることにする。例えば，（Ｖ［σ１］，Ｖ［σ２］，…，Ｖ［σｎ］）で１つの置換を表す。但し集合として｛１，２，…，ｎ｝＝｛σ１，σ２，…，σｎ｝である。

【０１０１】この時，σｎの値は、上記Ｒ＿ｎの値により定まることは明らかである。またσ（ｎ－１）の値も、Ｒ＿ｎ，Ｒ＿（ｎ－１）の値により定まり、σｎの値とは異なる全ての値から一様に選ばれることは明らかである。以下同様にして、σｉの値は、Ｒ＿ｎ，Ｒ＿（ｎ－１），…，Ｒ＿（ｉ＋１）の値により定まり、σｎ，σ（ｎ－１），…，σ（ｉ＋１）の値とは異なる全ての値から決まる。よって、上記配列要素置換部１２０の処理により、初期決定配列Ｖ１の全ての置換方法が決まっている。即ち、０≦Ｘ≦（ｎ！－１）なる整数Ｘと、初期決定配列Ｖ１の置換方法は１対１に対応していることが分かる。

【０１０２】今まで説明してきた議論は、初期決定配列Ｖ１の各要素の値が同じであるかどうかは関係なく、各要素の位置情報、即ちインデックスが違えば、違う要素と考えて行ってきた。しかし実際には、初期決定配列Ｖ１の各要素は、ｎ１個の１、ｎ２個の－１、（ｎ－ｎ１－ｎ２）個の０から構成されている。出力される配列Ｖにおいて、１が入っているｎ１個の要素の位置集合が同じであれば、配列Ｖの値も同じである。このことは、ｎ２個の－１、（ｎ－ｎ１－ｎ２）個の０についても同様のことが言える。以下では、どれくらい同じ出力配列を出すことがあるかについて説明していく。

【０１０３】次に、配列要素置換部１２０からの出力の一様性について説明する。先に述べたように、０≦Ｘ≦（ｎ！－１）を満たす整数Ｘと「ｎ個の元の置換」は１対１に対応する。その変換をτとする。配列要素置換部１２０は、整数Ｘに対応するτを用いて、τ（Ｖ１）をＶとして出力しているとになる。τからτ（Ｖ１）は一意的に決定する。次にあるτ０に対して、

【０１０４】τ０（Ｖ１）＝τ１（Ｖ１）となる変換τ１がいくつ存在するかについて説明する。τ０（Ｖ１）におけるｎ１個の１については、その位置が入れ替わっても、τ０（Ｖ１）の値としては変わらない。同様に、ｎ２個の－１についても、（ｎ－ｎ１－ｎ２）の０についても、その位置が入れ替わってもτ０（Ｖ１）の結果は変わらない。しかし、１入っている位置と０の入っている位置が入れ替わったり、１の入っている位置と－１の入っている位置が入れ替わったりすると、τ０（Ｖ１）の結果が変わってしまう。ゆえに、ｎ１個の１、ｎ２個の－１、（ｎ－ｎ１－ｎ２）個の０内の位置の入れ替えの組み合わせ分しか、取りうるτ１は存在しない。１がｎ１個であるため、（ｎ１）！種類ある。同様に、－１がｎ２個、０が（ｎ－ｎ１－ｎ２）個の入れ替えは、それぞれ、（ｎ２）！、（ｎ－ｎ１－ｎ２）！種類ある。従って、τ１は（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！種類存在する。

【０１０５】ゆえに、実施の形態１の配列出力部１００は、ｎ！種類の整数ＸをｎＣ！／（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）種類の置換に変換可能であることがいえる。また、一様性については、先の議論が置換の種類に関係なく、ｎ１、ｎ２により決まっていることは明らかである。

【０１０６】以上説明したように、配列出力部１００は、入力される整数Ｘに基づいて一様にｎ次元の配列を出力することができる。また、以上の説明から配列出力部１００は、同じ入力に対していつも同じ出力を行うことは明らかである。

【０１０７】以上は、Ｘ＜ｎ！の場合についての議論であるが、Ｘ＞ｎ！の場合もＸをｎ！で割った余りを取ることによって、同様の議論が可能である。

【０１０８】実際のパラメータの場合について説明する。ｎ＝２６３，ｎ１＝１６、ｎ２＝１６とする時、Ｌ（ｎ、ｎ１、ｎ２）に属する配列は、ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）≒２＾１６３種類存在する。ここで、ｘ＾ｙはｘのｙ乗を表している。ｎ！≒２＾１７４１であり、ハッシュ関数の分布の一様性を保持するためには、ハッシュ関数の出力長として、１７４１ビット以上必要になる。出力が１７４１ビット以上のハッシュ関数は現在のところ存在しないが、ハッシュ関数を用いて、ハッシュ関数の出力長より長いハッシュ関数値を計算する方法がある。そのため、出力が１７４１ビット以上のような長いハッシュ関数値が必要であっても、安全な暗号方式の様成することを妨げるような問題にはならない。

【０１０９】このように本実施の形態によれば、配列出力部１００は、入力された整数Ｘに基づいて一様にｎ次元の配列を出力することができる。そのため、ＮＴＲＵ暗号にＦＯＳＲＴを適用した場合において、配列出力部１００は、ハッシュ関数部４０より出力されるハッシュ関数値Ｈ（ｍ）を基に、一様にｎ次元の配列を出力することができ、ハッシュ関数部４０によるハッシュ関数の分布の一様性を保持することが可能となる。従って、暗号化装置１０は、安全性の高い暗号文ｃ（ｖ）を生成することができる。

【０１１０】さらに、配列出力部１００は、整数Ｘのみに基づいて配列Ｖを設定しているため、メモリテーブルを用いる必要がなく、メモリが少なくてよいという効果が得られる。

【０１１１】また、図１（ｂ）の復号化装置１５の配列出力部１０５もも配列出力部１００と同様の様成をしており、暗号化装置１０が暗号化した暗号文の復号が可能となる。

【０１１２】尚、本実施の形態では、暗号化装置１０および復号化装置１５を様成する各部が行う処理は、マイ

クロコンピュータのソフトウェアで行うものとしたが、電子回路やＩＣなどのハードウェアで動作するようにしてもよい。

【０１１３】また、配列出力部１００を暗号化装置１０に用いて、ハッシュ関数値を基に配列を出力する構成としたがこれに限定されない。

【０１１４】さらに、図２に示した配列出力部１００は、初期配列決定部１１０と配列要素置換部１２０とを備え、初期配列決定部１１０が決定した初期決定配列Ｖ１の各要素を配列要素置換部１２０が整数Ｘを基に置換する構成であるが、整数Ｘと予め設定された初期決定配列Ｖ１を入力とし、その初期決定配列Ｖ１を、整数Ｘを基に置換して配列Ｖを出力する配列要素置換部１２０と同様の処理を行う構成の配列出力部（以下、配列出力部１００ａという）であってもよい。

【０１１５】このような構成の配列出力部１００ａは、配列要素置換部１２０と同様の処理を行うため、ハッシュ関数の分布の一様性を保持することができ、さらにテーブルを用いず、整数Ｘのみの情報より配列Ｖ１を置換しているため、メモリが少なくてよいという効果が得られる。

【０１１６】また、配列出力部１００ａは、整数Ｘを鍵、初期決定配列Ｖ１をメッセージ、配列Ｖを暗号文とする暗号装置または、暗号方法であってもよい。また、配列出力部１００ａを使用する暗号装置または、暗号方法であってもよい。

【０１１７】（実施の形態２）以下、本発明における第２の実施の形態に係る暗号化装置について説明する。本実施の形態における暗号化装置は、図１（ａ）の暗号化装置１０と比べて配列出力部１００が異なる構成の配列出力部２００となっている。その他の構成は共通しているため、その説明を省略する。

【０１１８】図５は、本実施の形態における配列出力部２００の構成を示すブロック図である。この配列出力部２００は、整数Ｘを入力とし、Ｌ（ｎ、ｎ１、ｎ２）に属する配列Ｖ２０を出力するものである。ここで、Ｌ（ｎ，ｎ１，ｎ２）は、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元配列全体であり、ｎ、ｎ１、ｎ２は配列出力部２００に予め設定されているものとする。

【０１１９】配列出力部２００は、第１の数配置部２１０と第２の数配置部２２０からなり、配列出力部１００と同様にマイクロコンピュータのソフトウェア、あるいは電子回路などのハードウェアにより処理を実行する。

【０１２０】第１の数配置部２１０は、整数Ｘを入力とし、ｎ１個の要素が１、その他の要素が０であるｎ次元の配列Ｖ１０と、整数Ｘに所定の演算が行われた整数Ｘ１を第２の数配置部２２０に出力する。第１の数配置部２１０は、すべての配列要素が０の配列要素を暫定的に決定し、０の配列要素を整数Ｘに基づいて１に改編する。

【０１２１】第２の数配置部２２０は、第１の数配置部２１０より出力された配列Ｖ１０と、整数Ｘ１とを入力とし、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元の配列Ｖ２０を出力する。ここで、第２の数配置部２２０は、第１の数配置部２１０が出力する配列の００の配列要素を－１に改編する。

【０１２２】まず、第１の数配置部２１０の動作を説明する。図６は、第１の数配置部２１０が行う処理を示すフローチャートである。第１の数配置部２１０は、以下のようなステップの処理で行われる。なお、以下では、配列Ｖ１０の（左から）ｉ番目の要素をＶ１０［ｉ］とする。また、カウンタｃ１’の値をカウント値ｃ１、カウンタｃ２’の値をカウント値ｃ２とする。

【０１２３】図７は第１の数配置部２１０における配列Ｖ１０の各段階での配列状態を示している。まず、第１の数配置部２１０は、整数Ｘを変数Ｙ１に代入する（ステップＳ２０１）。

【０１２４】次に、第１の数配置部２１０は、配列Ｖ１０を全ての要素を０（整数Ｐ１）とする（ステップＳ２０２）。ここで、初期配列が決定される。次に、第１の数配置部２１０は、カウンタｃ１’のカウント値ｃ１を１に設定する（ステップＳ２０３）。

【０１２５】次に、第１の数配置部２１０は、カウンタｃ２’のカウント値ｃ２をｎに設定する（ステップＳ２０４）。次に、第１の数配置部２１０は、変数Ｙ１（被除数）をカウント値ｃ２（除数）で割った商Ｓと余りＲを求める（ステップＳ２０５）。

【０１２６】次に、第１の数配置部２１０は、配列Ｖ１００の０である要素の中で、左から（Ｒ＋１）番目の要素を１（整数Ｐ２）に設定する（ステップＳ２０６）。次に、第１の数配置部２１０は、商Ｓを変数Ｙ１に代入する（ステップＳ２０７）

【０１２７】次に、第１の数配置部２１０は、カウント値ｃ１＝ｎ１であるかを判定し（ステップＳ２０８）、カウント値ｃ１＝ｎ１でないと判定した場合は（ステップＳ２０８のＮｏ）、配列Ｖ１０の１である要素がｎ１個に達していないとして、カウント値ｃ１をカウントアップし（ｃ１←ｃ１＋１）、カウント値ｃ２をカウントダウン（ｃ２←ｃ２－１）する処理に移る（ステップＳ２０９）。そして、第１の数配置部２１０は、再度、変数Ｙ１をカウント値ｃ２で割った商Ｓと余りＲを求める処理（ステップＳ２０５）を行う。

【０１２８】一方、第１の数配置部２１０は、カウント値ｃ１＝ｎ１であると判定した場合は（ステップＳ２０８のＹｅｓ）、配列Ｖ１００の１である要素がｎ１個になったとして、配列Ｖ１０と、変数Ｙ１（整数Ｘ１）の値を第２の数配置部２２０に出力して（ステップＳ２１０）、処理を終了する。

【０１２９】このように、第１の数配置部２１０は、カウンタｃ１’のカウント値ｃ１がｎ１となるまで、商Ｓと余りＲを求める処理（ステップＳ２０５）からカウント値ｃ１をカウントアップし、カウント値ｃ２をカウントダウンする処理（ステップＳ２０９）までを繰り返す。そして、カウンタｃ１’のカウント値ｃ１がｎ１となると、すなわち配列Ｖ１０の要素のうち１がｎ１個となると、第１の数配置部２１０は、配列Ｖ１０を第２の数配置部２２０に出力する。

【０１３０】この第１の数配置部２１０の動作を、具体例を用いて説明する。実際に入力される整数Ｘ＝５６４４とし、また第１の数配置部２１０が出力するｎ次元の配列Ｖ１０が、例えば８次元の配列であり（ｎ＝８）、３個（ｎ１＝３）の要素が１、その他の５個の要素が０である場合の例を用いて説明する。

【０１３１】まず、第１の数配置部２１０は、変数Ｙ１に５６４４を代入する（ステップＳ２０１）。次に、第１の数配置部２１０は、配列Ｖ１０の配列状態を、図７の配列Ｖ１１に示す、すべての要素が０である配列状態とする（ステップＳ２０２）。

【０１３２】次に、第１の数配置部２１０は、カウント値ｃ１＝１とし（ステップＳ２０３）、カウント値ｃ２＝８（ステップＳ２０４）とする。次に、第１の数配置部２１０は、変数Ｙ１＝５６４４、およびカウント値ｃ２＝８より、商Ｓ＝７０５、余りＲ＝４を求める（ステップＳ２０５）。

【０１３３】次に、第１の数配置部２１０は、配列Ｖ１０の０である要素のうち左から（Ｒ＋１）番目、すなわち５番目の要素Ｖ１０［５］を１に設定し、図７の配列Ｖ１２に示す配列状態とする（ステップＳ２０６）。次に、第１の数配置部２１０は、変数Ｙ１に７０５（＝商Ｓ）を代入する（ステップＳ２０７）。

【０１３４】そして、カウント値ｃ１＝１であり、カウント値ｃ１＝３（＝ｎ１）でないので（ステップＳ２０８のＮｏ）、第１の数配置部２１０は、カウント値ｃ１＝２（カウントアップ）、カンウト値ｃ２＝７（カウントダウン）とする（ステップＳ２０９）。

【０１３５】次に、第１の数配置部２１０は、変数Ｙ１＝７０５、およびカウント値ｃ２＝７より、再度、商Ｓ、余りＲを求める（ステップＳ２０５）。商Ｓ＝１００、余りＲ＝５となる。

【０１３６】次に、第１の数配置部２１０は、配列Ｖ１０の０である要素のうち左から６番目の要素を１に設定する。この設定前の配列Ｖ１０の配列状態は、図７の配列Ｖ１２に示す配列状態である。配列Ｖ１２の０の要素は、Ｖ１２［５］以外の要素である。配列Ｖ１２の０の要素の中で６番目の要素は、Ｖ１２［７］であるので、Ｖ１２［７］の０を１にする（ステップＳ２０６）。これにより、図７に示す配列Ｖ１３となる。

【０１３７】このように、第１の数配置部２１０は、配列Ｖ１０の要素のうち１がｎ１個（本例ではｎ１＝３）となるまで、余りＲに応じて配列Ｖ１０の０の要素を１にし、図７の配列Ｖ１４に示すように１である要素が３個となれば、その配列を配列Ｖ１０として第２の数配置部２２０に出力する。

【０１３８】次に、第２の数配置部２２０の動作を説明する。図８は、第２の数配置部２２０が行う処理を示すフローチャートである。図９は第２の数配置部２２０における配列Ｖ２０の各段階での配列状態を示している。なお、以下では、配列Ｖ２０の（左から）ｉ番目の要素をＶ１０［ｉ］とする。また、カウンタｃ１’の値をカウント値ｃ１、カウンタｃ２’の値をカウント値ｃ２とする。

【０１３９】まず、第２の数配置部２２０は、第１の数配置部２１０から出力された変数Ｙ１（整数Ｘ１）の値を変数Ｙ２に代入する（ステップＳ３０１）。次に、第２の数配置部２２０は、第１の数配置部２１０から出力された配列Ｖ１０を配列Ｖ２０に代入する（ステップＳ３０２）。次に、第２の数配置部２２０は、カウント値ｃ１を１に設定する（ステップＳ３０３）。

【０１４０】次に、第２の数配置部２２０は、カウント値ｃ２を（ｎ－ｎ１）に設定する（ステップＳ３０４）。次に、第２の数配置部２２０は、変数Ｙ２（被除数）をカウント値ｃ２（除数）で割った商Ｓと余りＲを求める（ステップＳ３０５）。次に、第２の数配置部２２０は、配列Ｖ２０の０である要素の中で、左から（Ｒ＋１）番目の要素を－１に設定する（ステップＳ３０６）。次に、第２の数配置部２２０は、商Ｓを変数Ｙ２に代入する（ステップＳ３０７）。

【０１４１】次に、第２の数配置部２２０は、カウント値ｃ１＝ｎ２であるかを判定し（ステップＳ３０８）、カウント値ｃ１＝ｎ２でないと判定した場合は（ステップＳ３０８のＮｏ）、配列Ｖ２０の－１である要素がｎ２個に達していないとして、カウント値ｃ１をカウントアップし（ｃ１←ｃ１＋１）、カウント値ｃ２をカウントダウン（ｃ２←ｃ２－１）する処理（ステップＳ３０９）に移る。つづいて、第２の数配置部２２０は、再度、変数Ｙ２をカウント値ｃ２で割った商Ｓと余りＲを求める処理を行う（ステップＳ３０５）。

【０１４２】一方、第２の数配置部２２０は、カウント値ｃ１＝ｎ２であると判定した場合は（ステップＳ３０８のＹｅｓ）、配列Ｖ２０の－１である要素がｎ２個になったとして、配列Ｖ２０を出力して（ステップＳ３１０）、処理を終了する。

【０１４３】このように、第２の数配置部２２０は、カウント値ｃ１＝ｎ２となるまで、商Ｓと余りＲを求める処理（ステップＳ３０５）からカウント値ｃ１をカウントアップし、カウント値ｃ２をカウントダウンする処理（ステップＳ３０９）までを繰り返し、カウント値ｃ１＝ｎ２となると、配列Ｖ２０の要素のうち－１がｎ２個

（本例ではｎ２＝２）となったとして、配列Ｖ２０を出力する。

【０１４４】この第２の数配置部２２０の動作を、具体例を用いて説明する。実際に第１の数配置部２１０から出力される整数Ｘ１＝１５０とし、また第２の数配置部２２０が出力するｎ次元の配列Ｖ２０が、例えば８次元の配列であり（ｎ＝８）、３個（ｎ１＝３）の要素が１であり、２個（ｎ２＝２）の要素が－１であり、その他の３個の要素が０である例を用いて説明する

【０１４５】まず、第２の数配置部２２０は、変数Ｙ２に１５０を代入する（ステップＳ３０１）。次に、第２の数配置部２２０は、配列Ｖ２０に、第１の数配置部２１０から出力される３個の要素が１であり、その他の５個の要素が０である配列Ｖ１０を代入する。図９の配列Ｖ２１がその代入した配列である（ステップＳ３０２）。

【０１４６】次に、第２の数配置部２２０は、カウント値ｃ１＝１とし（ステップＳ３０３）、カウント値ｃ２＝ｎ－ｎ１＝８－３＝５とする（ステップＳ３０４）。次に、第２の数配置部２２０は、変数Ｙ２＝１５０、およびカウント値ｃ２＝５より、商Ｓ＝３０、余りＲ＝０を求める（ステップＳ３０５）。

【０１４７】次に、第２の数配置部２２０は、配列Ｖ２０の０である要素のうち左から１番目の要素、すなわちＶ２０［１］を－１にする（ステップＳ３０６）。図９の配列Ｖ２２がその配列状態を示している。次に、第２の数配置部２２０は、変数Ｙ２＝３０（＝商Ｓ）とする（ステップＳ３０７）。

【０１４８】そして、カウント値ｃ１＝１であり、カウント値ｃ１はｎ２（＝２）でないので（ステップＳ３０８のＮｏ）、第２の数配置部２２０は、カウント値ｃ１＝２（カウントアップ）、カウント値ｃ２＝４（カウントダウン）とする（ステップＳ３０９）。

【０１４９】次に、第２の数配置部２２０は、変数Ｙ２＝３０、およびカウント値ｃ２＝４より、再度、商Ｓ、余りＲを求める（ステップＳ３０５）。商Ｓ＝７、余りＲ＝２となる。

【０１５０】次に、第２の数配置部２２０は、配列Ｖ２０の０である要素のうち左から（Ｒ＋１）番目、すなわち３番目の要素を－１に設定する。設定前の配列Ｖ２０の配列状態は、図９に示す配列Ｖ２２であり、０の要素は、Ｖ２２［３］、Ｖ２２［４］、Ｖ２２［６］、Ｖ２２［８］である。配列Ｖ２２の０の要素の中で左から３番目の要素は、Ｖ２２［６］であるので、Ｖ２２［６］の０を－１にする（ステップＳ３０６）。この設定後の配列状態は、図９に示す配列Ｖ２３である。

【０１５１】次に、第２の数配置部２２０は、変数Ｙ２に７（＝商Ｓ）を代入する（ステップＳ３０７）。

【０１５２】そして現時点で、カウント値ｃ１＝２、即ちカウント値ｃ１＝ｎ２であるため（ステップＳ３０８のＹｅｓ）、第２の数配置部２２０は、配列Ｖ２０を出力する（ステップＳ３１０）。この出力される配列２０は、図９の配列Ｖ２３に示す配列状態である。

【０１５３】このように、配列出力部２００は、ハッシュ関数値Ｈ（ｍ）である整数Ｘからｎ１個の要素が１、ｎ２個の要素が－１、（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元の配列を出力する。

【０１５４】上述した配列出力部２００は、０≦Ｘ≦（（（ｎ！）／（ｎ－ｎ１－ｎ２）！）－１）を満たす整数Ｘに基づいて、一様な配列Ｌ（ｎ、ｎ１、ｎ２）を出力している。以下では、０≦Ｘ≦（（（ｎ！）／（ｎ－ｎ１－ｎ２）！）－１）を満たす整数Ｘに限定して説明を行う。

【０１５５】第１の数配置部２１０では、整数Ｘに基づいた位置の配列Ｖ１０の要素を１に設定している。以下では、この要素を１に設定する処理について具体的に見ていく。また、実施の形態１と同様に、第１の数配置部２１０の構成において、カウント値ｃ２におけるステップＳ２０５の余りをＲ＿ｃ２とする。全体の説明の流れは、以下の通りである。

（１）入れ替え処理が重複しないことについて説明する。

（２）（１）の結果を利用して、数Ｘと配列要素置換部における処理内容が１対１に対応していることを説明する。

（３）同じ配列を出力する数Ｘが何種類あるかを計算する。

【０１５６】この（３）の計算において配列に依らず、整数Ｘの種類の数が同じであることが示すことが可能であるため、入力に対して出力が一様に分布することがいえる。

【０１５７】まず、配置処理が重複しないことについて説明する。以下では、まず、配列の同じ位置に対する値（１または－１）の配置が複数回起こらない、即ち、配置が終わった位置は、その処理以降の配置の対象にならないことについて説明する。

【０１５８】・カウント値ｃ２がｎの時

ステップＳ２０６では、配列Ｖ１０の０である要素の中で、Ｒ＿ｎ＋１番目の要素を１に設定している。ここで、カウント値ｃ２がｎの時は、Ｒ＿ｎ＋１番目の要素を１に設定する前のＶ１０は全ての要素が０であるため、単純に、Ｒ＿ｎ＋１番目を１に設定していることになる。

【０１５９】・カウント値ｃ２がｎ－１の時

ステップＳ２０６では、配列Ｖ１０の０である要素の中で、Ｒ＿（ｎ－１）＋１番目の要素を１に設定している。ここで、配列Ｖ１０の０である要素を１に設定する要素の対象としているため、カウント値ｃ２がｎの時に設定したＲ＿ｎ番目の要素はその対象とならないことに注意する。配列Ｖ１０が０である要素の中で、Ｒ＿（ｎ

－１）＋１番目の要素は、Ｒ＿（ｎ－１）＜Ｒ＿ｎである時は、単純に、配列Ｖ１０のＲ＿（ｎ－１）＋１番目の要素になっている。Ｒ＿（ｎ－１）＞Ｒ＿ｎである時は、配列Ｖ１０のＲ＿（ｎ－１）＋２番目の要素になっている。

【０１６０】上記のように、カウント値ｃ２がｉの時にｌに設定した要素は、カウント値ｃ２がｉ＋１以降のステップで、ｌに設定する要素の対象にならないため、１回設定した要素をさらに設定することがない。

【０１６１】・第１の数配置部２１０の処理が終了した後

従って、カウント値ｃ２がｎから（ｎ－ｎ１＋１）までの全ての処理が終わった時点で、配列Ｖ１０は、ｎ１個の要素が１であり、その他の（ｎ－ｎ１）個の要素が０になっている。上記の第１の数配置部２１０の議論は、第２の数配置部２２０でも同様に行える。第２の数配置部２２０の処理が終了した時点で、配列Ｖ２０は、ｎ１個の要素が１であり、ｎ２個の要素が－１であり、その他の（ｎ－ｎ１－ｎ２）個の要素が０になっている。

【０１６２】第１の数配置部２１０と第２の数配置部の処理の一意性について説明する。以下では、上記の結果（値の配置が重複しない）を利用して、整数Ｘと数配置部における処理内容が１対１に対応していることを説明していく。

【０１６３】・整数Ｘと数列の対応

整数Ｘを０≦Ｘ≦（（（ｎ！）／（ｎ－ｎ１－ｎ２）！）－１）を満たす整数とする。この時、実施の形態１と同様に、カウント値ｃ２がｉの時の第１の数配置部のステップＳ２０５のＲをＲ＿ｉ、第２の数配置部の同様の処理を行うステップＳ３０５のＲもＲ＿ｉとする。ここで、第１の数配置部２１０のカウント値ｃ２は、ｎからｎ－ｎ１＋１までの範囲にあり、第２の数配置部２２０のカウント値ｃ２は、ｎ－ｎ１からｎ－ｎ１－ｎ２＋１までの範囲にあるため、カウント値ｃ２が第１の数配置部と、第２の数配置部の間で重なることがないことを注意しておく。この時、整数Ｘは、実施の形態１と同様に以下のようにおける。

【０１６４】Ｘ＝　（ｎ×（ｎ－１）×…×（ｎ－ｎ１－ｎ２＋２））×Ｒ＿（ｎ－ｎ１－ｎ２＋１）＋（ｎ×（ｎ－１）×…×（ｎ－ｎ１－ｎ２＋３））×Ｒ＿（ｎ－ｎ１－ｎ２＋２）・・・＋（ｎ×（ｎ－１））×Ｒ＿（ｎ－２）＋ｎ×Ｒ＿（ｎ－１）＋Ｒ＿ｎ

従って、整数Ｘは、数列Ｒ＿（ｎ－ｎ１－ｎ２＋１）、…、Ｒ＿ｎが１対１対応していることが分かる。

【０１６５】・数列と整数配置

Ｒ＿ｉ（ｎ－ｎ１－ｎ２＋１≦ｉ≦ｎ）によって、カウンタｃ２がｉの時に、配列Ｖ２０のＲ’＿ｉ＋１番目がｎ－ｎ１＋１≦ｉ≦ｎの場合は、１、ｎ－ｎ１－ｎ２＋１≦ｉ≦ｎ－ｎ１の場合は、－１に設定されたと考える。その時、数列Ｒ＿（ｎ－ｎ１－ｎ２＋１）、…、Ｒ＿ｎと、上記の設定は、順序即ち、１または、－１に設定された時のカウンタｃ２の値も含めて考えると１対１対応することが分かる。

【０１６６】カウント値ｃ２がｊ（ｊ≠ｉ）の時に、Ｒ’＿ｉ＋１番目の要素をｙ（ｙは１または、－１）に設定し、カウント値ｃ２がｉの時にＲ’＿ｊ＋１番目の要素をｙに設定するように、ｙに設定する時のカウント値ｃ２の値が入れ替わっても、同じ配列Ｖ２０を出力する。この他に同じ配列Ｖを出力するようなケースは存在しない。このようなカウント値ｃ２の値の入れ替えは、実施の形態１と同様に考えると、ｙが１の場合で、（ｎ１）！個、ｙが－１の場合で、（ｎ２）！個存在する。従って、同じ配列Ｖ２０を出力する整数Ｘは、（ｎ１）！×（ｎ２）！種類存在することになる。

【０１６７】Ｌ（ｎ、ｎ１、ｎ２）に属する配列は、ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）種類存在する。ゆえに、実施の形態２の配列出力部２００は、ｎ！／（ｎ－ｎ１－ｎ２）！種類の整数Ｘに基づいて、ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）種類の配列を一様に出力可能であることがいえる。

【０１６８】また、以上の説明から配列出力部２００は、同じ入力に対していつも同じ出力を行うことは明らかである。

【０１６９】以上は、Ｘ＜ｎ！／（ｎ－ｎ１－ｎ２）！の場合についての議論であるが、Ｘ＞ｎ！／（ｎ－ｎ１－ｎ２）！の場合もＸを（ｎ！／（ｎ－ｎ１－ｎ２）！）で割った余りを取ることにによって、同様の議論が可能である。

【０１７０】ここで、実施の形態２と実施の形態１の効果との差異を説明する。実施の形態１では、カウンタｃ’のカウント値ｏは、ｎから２までの範囲で動く。それに対して、実施の形態２では、第１の数配置部２１０と第２の数配置部２２０において、カウンタｃ２’のカウント値ｃ２は、ｎから（ｎ－ｎ１－ｎ２＋１）までの範囲で動く。そのため、整数Ｘの値が、実施の形態１では、ｎ！種類存在するのに対し、実施の形態２では、（ｎ！／（ｎ－ｎ１－ｎ２）！）種類存在する。ゆえに、実施の形態２は実施の形態１より、入力の種類が（１／（ｎ－ｎ１－ｎ２）！）に削減でき、必要な入力のビット長も短くてすむ。

【０１７１】実際のパラメータの場合について説明する。ｎ＝２６３、ｎ１＝１６、ｎ２＝１６とする時、Ｌ（ｎ、ｎ１、ｎ２）に属する配列は、ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）≒２＾１６３である。この時、ｎ！／（ｎ－ｎ１－ｎ２）！≒２＾２５５であり、ハッシュ関数の分布の一様性を保持するためには、ハッシュ関数の出力長として、２５５ビット以上必要になる。実施の形態１に比べると必要なハッシュ関数出力長が１４８６ビット小さく、実施の形態１より

効率がよい。

【０１７２】このように、本実施の形態によれば、配列出力部２００は、整数Ｘに基づいてｎ次元の配列を一様に出力することができる。そのため、ＮＴＲＵ暗号にＦＯＳＲＴを適用した場合において、図１（ａ）の暗号化装置１００の配列出力部１００の代わりにこの配列出力部２００とし、ハッシュ関数部４０より出力されるハッシュ関数値Ｈ（ｍ）に基づいて、配列出力部２００がｎ次元の配列を一様に出力することで、ハッシュ関数の分布の一様性を保持することが可能となり、暗号化装置１００で生成する暗号文の安全性を高めることができる。

【０１７３】さらに、配列出力部２００は、整数Ｘのみの情報より配列Ｖ２０を設定しているため、メモリテーブルを用いる必要がなく、メモリが少なくてよいという効果が得られる。

【０１７４】また、図１（ｂ）の復号化装置１５の配列出力部１０５の代わりに、この配列出力部２００を用いることで暗号化された暗号文の復号が可能となる。

【０１７５】尚、また、配列出力部２００を暗号化装置１０に用いて、ハッシュ関数値を基に配列を出力する構成としたがこれに限定されない。

【０１７６】（実施の形態３）本発明における第３の実施の形態に係る暗号化装置を説明する。本実施の形態の暗号化装置は、図１の暗号化装置１０と比べて配列出力部１００が異なる構成の配列出力部３００となっている。その他の構成は共通しているため、その説明を省略する。

【０１７７】　本実施の形態における配列出力部３００を、図面を用いて説明する。図１０は、本実施の形態における配列出力部３００の構成を示すブロック図である。

【０１７８】この配列出力部３００は、整数Ｘを入力とし、Ｌ（ｎ、ｎ１、ｎ２）に属する配列Ｖ４０を出力するものである。ここで、ｎ、ｎ１、ｎ２は予め設定され、この配列出力部３００に与えられているものとする。

【０１７９】配列出力部３００は、第１の数配置部３１０と第２の数配置部３２０からなり、配列出力部１００と同様、マイクロコンピュータのソフトウェア、あるいは電子回路などのハードウェアにより処理を実行する。

【０１８０】第１の数配置部３１０は、整数Ｘを入力とし、ｎ１個の要素が１、その他の要素が０であるｎ次元の配列Ｖ３０と、整数Ｘに所定の演算が行われた整数Ｘ２を出力する。第１の数配置部３１０は、すべての配列要素が０の配列要素を暫定的に決定し、０の配列要素を整数Ｘに基づいて１に改編する。

【０１８１】第２の数配置部３２０は、第１の数配置部３１０より出力された配列Ｖ３０および整数Ｘ２を入力として、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元の配列Ｖ４０を出力する。ここで、第２の数配置部３２０は、第１の数配置部３１０が出力する配列の０の配列要素を－１に改編する。

【０１８２】第１の数配置部３１０の動作を説明する。図１１は、第１の数配置部３１０の処理を示すフローチャートである。第１の数配置部３１０は、以下のようなステップの処理で行われる。なお、以下では、配列Ｖ３０の（左から）ｉ番目の要素をＶ３０［ｉ］とする。また、カウンタｃ１′の値をカウント値ｃ１、カウンタｃ２′の値をカウント値ｃ２とする。また、Ｃ（ｓ、ｔ）は、ｓ個のものからｔ個を選ぶ組み合わせの数を示す。具体的には、Ｃ（ｓ、ｔ）＝ｓ！／（（ｓ－ｔ）！×ｔ！）である。

【０１８３】まず、第１の数配置部３１０は、整数Ｘを変数Ｚ１に代入し、配列Ｖ３０を全ての要素が０のｎ次元配列とする（ステップＳ４０１）。ここで、初期配列が決定される。

【０１８４】次に、第１の数配置部３１０は、カウンタｃ１′のカウント値ｃ１をｎ１に設定し、カウンタｃ２′のカウント値ｃ２をｎ１に設定する（ステップＳ４０２）。

【０１８５】次に、第１の数配置部３１０は、変数Ｚ１≧Ｃ（ｎ－ｃ２，ｃ１）であるかを判定する（ステップＳ４０３）。第１の数配置部３１０は、変数Ｚ１≧Ｃ（ｎ－ｃ２、ｃ１）であると判定した時（ステップＳ４０３のＹｅｓ）、変数Ｚ１←Ｚ１－Ｃ（ｎ－ｃ２、ｃ１）を代入し、カウンタｃ１′のカウント値ｃ１をカウントダウンし（ｃ１←ｃ１－１）、配列Ｖ３０の（ｎ－ｃ２＋１）番目の要素をｉにする（Ｖ３０［ｎ－ｃ２＋１］←ｉ）（ステップＳ４０４）。そして、第１の数配置部３１０は、カウンタｃ２′のカウント値ｃ２をカウントアップする（ステップＳ４０６）。

【０１８６】一方、第１の数配置部３１０は、変数Ｚ１≧Ｃ（ｎ－ｃ２、ｃ１）でないと判定した時（ステップＳ４０３のＮｏ）、配列Ｖ３０の（ｎ－ｃ２＋１）番目の要素を０にする（Ｖ３０［ｎ－ｃ２＋１］←０）（ステップＳ４０５）。そして、第１の数配置部３１０は、カウンタｃ２′のカウント値ｃ２をカウントアップする（ステップＳ４０６）。

【０１８７】このように、変数Ｚ１とＣ（ｎ－ｃ２，ｃ１）の大小関係に応じて、配列Ｖ３０の（ｎ－ｃ２＋１）番目の要素を０か１に設定する。そして、第１の数配置部３１０は、カウント値ｃ２をカウントアップすると（ステップＳ４０６）、カウント値ｃ２＞ｎを判定する（ステップＳ４０７）。

【０１８８】第１の数配置部３１０は、カウント値ｃ２＞ｎでないと判定した場合は（ステップＳ４０７のＮｏ）、再度、変数Ｚ１≧Ｃ（ｎ－ｃ２、ｃ１）であるかを判定する判定処理を行い（ステップＳ４０３）、その判定処理（ステップＳ４０３）からカウント値ｃ２＞ｎ

を判定する（ステップＳ４０７）までの処理を、カウント値ｃ２＞ｎとなるまで繰り返す。

【０１８９】一方、第１の数配置部３１０は、カウンタｃ２＞ｎと判定した場合（ステップＳ４０７のＹｅｓ）、配列Ｖ３０と、整数Ｘ２（整数Ｘ２＝整数Ｘ／Ｃ（ｎ，ｎ１））とを第２の数配置部３２０に出力する。

【０１９０】この第１の数配置部３１０の動作を、具体例を用いて説明する。実際に入力される整数Ｘ＝５０とし、また第１の数配置部３１０が出力する配列Ｖ３０が、例えば８次元の配列であり（ｎ＝８）、４個（ｎ１＝４）の要素が１、その他の４個の要素が０である場合の例を用いて説明する。

【０１９１】図１３（ａ）は、配列Ｖ３０の各段階の配列状態を示している。

【０１９２】まず、第１の数配置部３１０は、変数Ｚ１に５０を代入する（ステップＳ４０１）。

【０１９３】次に、第１の数配置部３１０は、カウンタｃ１’のカウント値ｃ１をＡとし、カウンタｃ２’のカウント値ｃ２をＢとする（ステップＳ４０２）。

【０１９４】この場合、変数Ｚ１（＝５０）≧Ｃ（７，４）（＝３５）であるので（ステップＳ４０３のＹｅｓ）、第１の数配置部３１０は、変数Ｚ１←５０－３５＝１５、カウント値ｃ１＝３、配列Ｖ３０の左から８番目の要素Ｖ３０［８］＝１とする（ステップＳ４０４）。このときの配列Ｖ３０の配列状態は図１３に示す配列Ｖ３１である。

【０１９５】次に、第１の数配置部３１０は、カウント値ｃ２を２にカウントアップする（ステップＳ４０６）。ここで、カウント値ｃ２＞８でないため（ステップＳ４０７のＮｏ）、第１の数配置部３１０は配列Ｖ３０の出力を行わず、再度、変数ＺｌとＣ（ｎ－ｃ２，ｃ１）の大小関係を判定する（ステップＳ４０３）。

【０１９６】この場合、変数Ｚ１（＝１５）＜Ｃ（６，３）（＝２０）であるので（ステップＳ４０３のＮｏ）、第１の数配置部３１０は、配列Ｖ３０の左から７番目の要素Ｖ３０［７］＝０とする（ステップＳ４０５）。このときの配列Ｖ３０の配列状態は図１３に示す配列Ｖ３２である。

【０１９７】このように、変数ＺｌとＣ（ｎ－ｃ２，ｃ１）の大小関係に応じて、配列Ｖ３０の各要素に順番に１か０を設定していき、全要素について決定したときに、第１の数配置部３１０は、第２の数配置部３２０に配列Ｖ３０を出力する。

【０１９８】上記の処理は、Ｓｃｈａｌｋｖｉｊｋのアルゴリズムと呼ばれる。Ｓｃｈａｌｋｖｉｊｋのアルゴリズムについて、Ｓｃｈａｌｋｖｉｊｋ，“Ａｎ Ａｌｇｏｒｉｔｈｍ ｆｏｒ Ｓｏｕｒｃｅ Ｃｏｄｉｎｇ”、ＩＴ７２－１８，１９７２が詳しい。以下では、この文献を「非特許文献４」とよぶ。

【０１９９】次に、第２の数配置部３２０の動作を説明する。図１２は、第２の数配置部３２０の処理を示すフローチャートである。第２の数配置部３２０は、以下のようなステップの処理で行われる。

【０２００】なお、以下では、配列Ｖ４０の（左から）ｉ番目の要素をＶ４０［ｉ］とする。また、カウンタｃ１’の値をカウント値ｃｌ、カウンタｃ２’の値をカウント値ｃ２とする。

【０２０１】まず、第２の数配置部３２０は、第１の数配置部３１０から出力された整数Ｘ２を変数Ｚ２に代入し、配列Ｖ３０を配列Ｖ４０に代入し、さらに配列Ｗを全ての要素が０である（ｎ－ｎ１）次元の配列とする（ステップＳ５０１）。

【０２０２】次に、第２の数配置部３２０は、カウンタｃｌ’のカウント値ｃｌをｎ２、カウンタｃ２’のカウント値ｃ２を１とする（ステップＳ５０２）。

【０２０３】次に、第２の数配置部３２０は、変数Ｚ２≧Ｃ（ｎ－ｎｌ－ｃ２，ｃｌ）であるかを判定し（ステップＳ５０３）、Ｚ２≧Ｃ（ｎ－ｎｌ－ｃ２，ｃｌ）であると判定した時（ステップ５０３のＹｅｓ）、変数Ｚ２にＺ２－Ｃ（ｎ－ｎｌ－ｃ２，ｃｌ）を代入し、カウント値ｃｌをカウントダウンし（ｃｌ←ｃｌ－１）、配列Ｗの（ｎ－ｎｌ－ｃ２＋１）番目の要素を－１にする（ステップＳ５０４）。そして、第２の数配置部３２０は、カウント値ｃ２をカウントアップする（ステップＳ５０６）。

【０２０４】一方、第２の数配置部３２０は、Ｚ２≧Ｃ（ｎ－ｎｌ－ｃ２，ｃｌ）でないと判定した時（ステップ５０３のＮｏ）、配列Ｗの（ｎ－ｎｌ－ｃ２＋１）番目の要素を０とする（ステップＳ５０５）。そして、第２の数配置部３２０は、カウント値ｃ２をカウントアップする（ステップＳ５０６）。

【０２０５】このように、変数Ｚ２とＣ（ｎ－ｎｌ－ｃ２，ｃｌ）の大小関係に応じて、配列Ｗの（ｎ－ｎｌ－ｃ２＋１）番目の要素が０か－１に設定される。

【０２０６】そして第２の数配置部３２０は、カウント値ｃ２のカウントアップを行うと（ステップＳ５０６）、カウント値ｃ２がｃ２＞ｎ－ｎｌであるかを判定する（ステップＳ５０７）。第２の数配置部３２０は、ｃ２＞ｎ－ｎｌでないと判定した時（ステップＳ５０７のＮｏ）、変数Ｚ２≧Ｃ（ｎ－ｎｌ－ｃ２，ｃｌ）であるかを再度判定する判定処理を行い（ステップＳ５０３）、ｃ２＞ｎ－ｎｌとなるまで、上記判定処理（ステップＳ５０３）からｃ２＞ｎ－ｎｌの処理（ステップＳ５０７）を繰り返す。

【０２０７】すなわち、第２の数配置部３２０は、カウント値ｃ２が１から（ｎ－ｎｌ）の各場合において、変数Ｚ２とＣ（ｎ－ｎｌ－ｃ２，ｃｌ）の大小関係に応じて、配列Ｗの（ｎ－ｎｌ－ｃ２＋１）番目の要素が０か－１に設定される。

【０２０８】一方、第２の数配置部３２０は、ｃ２＞ｎ

－ｎ１であると判定すると（ステップＳ５０７のＹｅｓ）、カウント値ｃ１をｌ、カウント値ｃ２をｌとする処理に移る（ステップＳ５０８）。

【０２０９】次に、第２の数配置部３２０は、配列Ｖ４０のｃ１番目の要素Ｖ４０［ｃ１］がｌであるかの判定処理を行い（ステップＳ５０９）、Ｖ４０［ｃ１］＝ｌと判定したとき（ステップＳ５０９のＹｅｓ）、カウント値ｃ１をカウントアップし（ｃ１←ｃ１＋１）（ステップＳ５１２）、再度、上記判定処理（ステップＳ５０９）を行う。

【０２１０】一方、第２の数配置部３２０は、Ｖ４０［ｃ１］＝１でないと判定したとき（ステップＳ５０９のＮｏ）、配列Ｖ４０のｃ１番目の要素Ｖ４０［ｃ１］に配列Ｗのｃ２番目の要素Ｗ［ｃ２］を代入し、カウント値ｃ２をカウントアップ（ｃ２←ｃ２＋１）する（ステップＳ５１０）。

【０２１１】次に、第２の数配置部３２０は、カウント値ｃ２がｃ２＞ｎ－ｎ１であるかを判定し（ステップＳ５１１）、ｃ２＞ｎ－ｎ１でないと判定したとき（ステップＳ５１２のＮｏ）、カウント値ｃ１をカウントアップする処理（ｃ１←ｃ１＋１）（ステップＳ５１２）に移る。

【０２１２】一方、第２の数配置部３２０は、カウント値ｃ２がｃ２＞ｎ－ｎ１であると判定した時は、配列Ｖ４０を外部に出力して処理を終了する。

【０２１３】ここで、上述したカウント値ｃ１をｌ、カウント値ｃ２をｌとする処理（ステップＳ５０８）から配列Ｖ４０を出力する（ステップＳ５１２まで）までの処理では、Ｖ［ｃ１］のｏｏの要素にＷ［ｃ２］の要素を順番に代入している。

【０２１４】この第２の数配置部３２０の動作を、具体例を用いて説明する。実際に入力される整数Ｘ２＝２０とし、また第１の数配置部３２０が出力する配列Ｖ４０が、例えば８次元の配列であり（ｎ＝８）、４個（ｎ１＝４）の要素が１、２個（ｎ２＝２）の要素が－１、その他の２個の要素が０である場合の例を用いて説明する。

【０２１５】図１３（ｂ）は、配列Ｖ４０の各段階の配列状態を示しており、図１３（ｃ）は、配列Ｗの各段階の配列状態を示している。

【０２１６】まず、第２の数配置部３２０は、変数Ｚ２＝２０とし、配列Ｖ４０に例えば、第１の数配置部３１０から出力された図１３（ｂ）の配列Ｖ４１に示す配列状態を代入し、さらに配列Ｗをすべての要素が０の（ｎ－ｎ１）次元（４次元）の配列とする（ステップＳ５０１）。

【０２１７】次に、第２の数配置部３２０は、カウンタｃ１’のカウント値ｃ１を２に設定し、カウンタｃ２をｌに設定する（ステップｓ５０２）。

【０２１８】次に、第２の数配置部３２０は、変数Ｚ２とＣ（ｎ－ｎ１－ｃ２，ｃ１）の大小関係を判定し（ステップＳ５０３）、変数Ｚ２（＝２０）≧Ｃ（３，２）（＝３）であるため（ステップＳ５０３のＹｅｓ）、第２の数配置部３２０は、変数Ｚ２＝２０－３＝１７とし、カウント値ｃ１をｌとし、配列Ｗの左から４番目の要素、すなわちＷ［ｎ－ｎ１－ｃ２＋１］を－１とする（ステップＳ５０４）。この配列状態は図１３（ｃ）の配列Ｗｌに示した通りである。

【０２１９】このように、配列Ｗの各要素に０か－１を設定する。図１３（ｃ）の配列Ｗ２は、全要素の設定された配列状態の一例である。

【０２２０】そして、第２の数配置部３２０は、配列Ｗの全要素を設定すると（ステップＳ５０７のＹｅｓ）、カウンタｃ１’のカウント値ｃ１をｌにし、カウンタｃ２’のカウント値ｃ２をｌにする（ステップＳ５０８）。この以降の処理で、配列Ｖ４１のｏｏの要素に配列Ｗの各要素を代入する。

【０２２１】現時点での配列Ｖ４０の配列状態を配列Ｖ４１とし、配列Ｗの配列状態を配列Ｗ２とする。まず、配列Ｖ４１のｃ１番目の要素、すなわちＶ４１［１］＝０であるため（ステップＳ５０９のＮｏ）、Ｖ４１［１］に配列Ｗのｃ２番目の要素、すなわちＷ［１］であるｏを代入する。図１３（ｂ）の配列Ｖ４２がその配列状態である。

【０２２２】このように、第２の数配置部３２０は、配列Ｖ４１のｏｏの要素に配列Ｗ２の全要素を代入し（ステップＳ５０９からステップＳ５１２のＳ処理）、配列４１を出力する。図１３（ｂ）の配列Ｖ４３が配列Ｖ４１のｏｏの要素に配列２の要素を代入した配列状態である。

【０２２３】ここで、配列出力部３００全体の動作を説明する。まず、第１の数配置部３１０は、整数Ｘを入力とし、ｎ１個の要素が１、他の要素が０であるn次元の配列Ｖ３０と、整数Ｘ２（整数Ｘ２＝整数Ｘ／Ｃ（ｎ，ｎ１））とを第２の数配置部３２０に出力する。

【０２２４】次に、第２の数配置部３２０は、第１の数配置部３１０から出力された配列Ｖ３０、整数Ｘ２を入力とし、ｎ１個の要素が１、ｎ２個の要素が－１、他の要素が０であるn次元の配列Ｖ４０を出力する。

【０２２５】本実施の形態では、Ｓｃｈａｌｋｖｉｊｋのアルゴリズムを応用している。具体的には、第１の数配置部３１０での配列Ｖ３０におけるｌの要素の配置場所を決定する箇所と、第２の数配置部３２０での配列Ｖ４０におけるｌの要素の配置場所を決定する箇所に、Ｓｃｈａｌｋｖｉｊｋのアルゴリズムを使用している。

【０２２６】この配列出力部３００は、０≦Ｘ≦Ｃ（ｎ，ｎ１）×Ｃ（ｎ－ｎ１，ｎ２）（＝ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）を満たす整数ＸをＬ（ｎ，ｎ１，ｎ２）に１対１に変換している。

【０２２７】以下では、０≦Ｘ≦Ｃ（ｎ，ｎ１）×Ｃ

（ｎ－ｎ１、ｎ２）を満たす整数Ｘに限定して、１対１に変換していることについて説明を行う。

【０２２８】第１の数配置部３１０、第２の数配置部３２０では、Ｓｃｈａｌｋｖｉｊｋアルゴリズムを利用して、ｌ、－ｌの配置場所を決定している。Ｓｃｈａｌｋｖｉｊｋアルゴリズムは、入力の出力の種類以下に限定すれば、１対１に変換可能であることが知られている。これについては、先の非特許文献４が詳しい。従って、実施の形態３における配列出力部３００は、整数Ｘに対して１対１に対応する配列Ｖ４０を出力している。そのため、配列出力部３００は、入力された整数Ｘに基づいて一様に配列を出力していることになる。

【０２２９】ここで、実施の形態３と、実施の形態１、２の効果との差異を説明する。整数Ｘの値が、実施の形態１では、一つの出力値に対して、入力値が（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！、実施の形態２では、（ｎ１）！×（ｎ２）！種類存在する。それに対して、実施の形態３では、１対１で変換する。そのため、入力の一様性を保持するために必要な入力のビット長が最小になっている。

【０２３０】実際のパラメータの場合について説明する。ｎ＝２６３、ｎ１＝１８、ｎ２＝１６とする時、φ（ｎ、ｎ１、ｎ２）に属する配列は、ｎ！／（（ｎ１）！×（ｎ２）！×（ｎ－ｎ１－ｎ２）！）≒２＾１６３であるので、ハッシュ関数の出力長は、１６３ビット以上でよく、これは、実施の形態２に比べて、９２ビット短い。

【０２３１】しかし、実施の形態３の配列出力装置３００の第１の数配置部、第２の数配置部では、Ｃ（ｎ－ｎ１、ｃ２）やＣ（ｎ－ｎ１－ｎ２、ｃ２）の計算を行う必要がある。この計算は、階乗の計算を含むため、計算量が大きくなる。一方、実施の形態１、２では、階乗の計算を行わないため、計算量が小さい。

【０２３２】このように、本実施の形態によれば、配列出力部３００は、整数Ｘに基づいて一様にｎ次元の配列を出力することができる。そのため、ＮＴＲＵ暗号にＦＯＳＲＴを適用して、図１（ａ）の暗号化装置１０の配列出力部１００の代わりにこの配列出力部３００とし、ハッシュ関数部４０より出力されるハッシュ関数値Ｈ（ｍ）に基づいて、配列出力部３００が一様にｎ次元の配列を出力することで、ハッシュ関数の分布の一様性を保持することが可能となり、暗号化装置１０で生成する暗号文の安全性を高めることができる。

【０２３３】さらに、配列出力部３００は、整数Ｘのみの情報より配列Ｖ４０を設定しているため、メモリテーブルを用いる必要がなく、メモリが少なくてよいという効果が得られる。

【０２３４】また、図１（ｂ）の復号化装置１５の配列出力部１０５の代わりに、この配列出力部３００を用いることで暗号化された暗号文の復号が可能となる。

【０２３５】尚、また、配列出力部３００を暗号化装置１０に用いて、ハッシュ関数値を基に配列を出力する構成としたがこれに限定されない。

【０２３６】（実施の形態４）本発明における第４の実施の形態に係る暗号化装置を説明する。本実施の形態の暗号化装置は、図１の暗号化装置１０と比べて配列出力部１００が異なる構成の配列出力部４００となっている。その他の構成は共通しているため、その説明を省略する。

【０２３７】本実施の形態における配列出力部４００を、図面を用いて説明する。図１４は、本実施の形態における配列出力部４００の構成を示すブロック図である。

【０２３８】この配列出力部４００は、整数Ｘを入力とし、Ｌ（ｎ、ｎ１、ｎ２）に属する配列Ｖ５０を出力するものである。ここで、Ｌ（ｎ，ｎ１，ｎ２）は、ｎ１個の要素が１、ｎ２個の要素が－１、その他の（ｎ－ｎ１－ｎ２）個の要素が０であるｎ次元配列全体であり、ｎ、ｎ１、ｎ２は配列出力部４００に予め設定されているものとする。この配列出力部４００は、すべての配列要素が０の配列要素を暫定的に決定し、０の配列要素を整数Ｘに基づいて１、および－１に改編する。

【０２３９】配列出力部４００は、配列出力部１００と同様、マイクロコンピュータのソフトウェア、あるいは電子回路などのハードウェアにより処理を実行する。

【０２４０】次に、配列出力部４００の動作を説明する。まず、配列出力部４００は、配列Ｖ５０をすべての要素が０である配列状態にする。

【０２４１】次に、配列出力部４００は、整数Ｘを８ビット毎に分割する。整数Ｘは０，１の２値で表されたビット情報の集まりで示されており、図１６に示すように、整数Ｘは８ビット毎に（ｎ１＋ｎ２）個に分割される。

【０２４２】図１６は、整数Ｘを各分割情報［０］，分割情報Ｄ［１］～分割情報Ｄ［ｎ１＋ｎ２－１］に分解した状態を示す図である。各分割情報Ｄ［０］，分割情報Ｄ［１］～分割情報Ｄ［ｎ１＋ｎ２－１］は、それぞれ８ビットの情報により整数を示す。

【０２４３】ここで、８ビットの分割情報Ｄ［０］が整数Ｑを示しているとすると、次に配列出力部４００は、配列Ｖ５０において、Ｑ＋１番目（以下、ｐ０番目という）の要素が０である場合に１に設定する。つづいて、配列出力部４００は、ｐ１＝（ｐ０＋Ｄ［１］）ｍｏｄ（ｎ）で示されるｐ１番目の要素が０である場合に１に設定する。

【０２４４】このように、配列出力部４００は、分割情報Ｄ［ｉ］を基に、配列Ｖ５０のｐｉ＝（Ｐ（ｉ－１）＋Ｄ［ｉ］）ｍｏｄ（ｎ）（ここで、ｉ＝１～ｎ１＋ｎ２－１）で示されるｐｉ番目の要素が０である場合に順に１に設定する。また、配列Ｖ５０のｐｉ番目の要素と

は配列Ｖ５０の左からｐｉ番目の要素をいう。

【０２４５】このとき、配列出力部４００は、配列Ｖ５０のｐｉ番目の要素が０でない場合は１に設定せず、ｐｉ←（ｐｉ＋１）ｍｏｄ（ｎ）とし、ｐｉ番目の右側にあるＯの要素を１に設定する。

【０２４６】配列出力部４００は、このように配列Ｖ５０の要素を１に設定する処理を行い、１の要素がｎ１個になると、次に同様に分割情報Ｄ［ｊ］を基に、配列Ｖ５０の－１の要素がｎ２個になるまで－１の要素を設定する処理を行う。

【０２４７】ここで配列出力部４００の詳細な動作を説明する。図１５は、配列出力部４００が行う処理を示すフローチャートである。配列出力部４００は、以下のようなステップの処理で行われる。なお、カウンタｃ１’の値をカウント値ｃ１、カウンタｃ２’の値をカウント値ｃ２とする。

【０２４８】まず配列出力部４００は、変数Ｙ１０に整数Ｘを代入する（ステップＳ６０１）。次に、配列出力部４００は、配列Ｖ５０を全ての要素が０である配列状態とする（ステップＳ６０２）。

【０２４９】次に、配列出力部４００は、整数Ｘを８ビット毎に区切って分割情報Ｄ［０］、分割情報Ｄ［１］～分割情報Ｄ［ｎ１＋ｎ２－１］に分割する（ステップＳ６０３）。

【０２５０】次に、配列出力部４００は、カウンタｃ１’のカウント値ｃ１を０にする（ステップＳ６０４）。

【０２５１】次に、配列出力部４００は、カウンタｃ２’のカウント値ｃ２をＤ［０］＋１とする（ステップＳ６０５）。すなわち、カウント値ｃ２は、８ビットの分割情報Ｄ［０］で示される整数Ｑ＋１の値となる。

【０２５２】次に、配列出力部４００は、配列Ｖ５０のｃ２番目の要素Ｖ５０［ｃ２］が０であるかを判定し（ステップＳ６０６）、０でないと判定した時は（ステップＳ６０６のＮｏ）、カウント値ｃ２をｃ２←（ｃ２＋１）ｍｏｄ（ｎ）とし（ステップＳ６０７）、再度、要素Ｖ５０［ｃ２］が０であるか否かの判定を行う（ステップＳ６０６）。一方、配列出力部４００は、Ｖ５０［ｃ２］が０であると判定した時は（ステップＳ６０６のＹｅｓ）、要素Ｖ５０［ｃ２］を１に設定する（ステップＳ６０８）。

【０２５３】これらの要素Ｖ５０［ｃ２］が０であるかの判定処理（ステップＳ６０６）、ｃ２←（ｃ２＋１）ｍｏｄ（ｎ）の処理（ステップＳ６０７）、およびＶ５０［ｃ２］←１の処理（ステップＳ６０８）により、Ｖ５０［ｃ２］の要素が０でない場合に、Ｖ５０［ｃ２］の右側の要素で０であるものを巡回移動しながら順に探して１に設定する。

【０２５４】そして次に、配列出力部４００は、カウンタｃ１’のカウント値ｃ１がｃ１＜ｎ１－１であるかの判定を行う（ステップＳ６０９）。

【０２５５】配列出力部４００は、ｃ１＜ｎ１－１であると判定したとき（ステップＳ６０９のＹｅｓ）、カウント値ｃ１をカウントアップすると共に（ｃ１←ｃ１＋１）、カウント値ｃ２をｃ２←（ｃ２＋Ｄ［ｃ１］）ｍｏｄ（ｎ）とし（ステップＳ６１０）、そして再度、Ｖ５０［ｃ２］が０であるかの判定を行う（ステップＳ６０６）。

【０２５６】このように、配列Ｖ５０の１の要素がｎ１個になるまで、１の要素の設定処理が行われる。

【０２５７】一方、配列出力部４００は、ｃ１＜ｎ１－１でないと判定したとき（ステップＳ６０９のＮｏ）、配列Ｖ５０の１の要素がｎ１個になったとして、配列Ｖ５０におけるー１の要素の設定処理に移るべく、カウント値ｃ１を０とし（ステップＳ６１１）、カウント値ｃ２をｃ２←（ｃ２＋Ｄ［ｎ１］）ｍｏｄ（ｎ）とする（ステップＳ６１２）。

【０２５８】次に、配列出力部４００は、配列Ｖ５０の要素Ｖ５０［ｃ２］が０であるかを判定し（ステップＳ６１４）、０でないと判定した時は（ステップＳ６１４のＮｏ）、カウント値ｃ２をｃ２←（ｃ２＋１）ｍｏｄ（ｎ）とし（ステップＳ６１３）、再度、要素Ｖ５０［ｃ２］が０であるか否かの判定を行う（ステップＳ６１４）。一方、配列出力部４００は、Ｖ５０［ｃ２］が０であると判定した時（ステップＳ６１４のＹｅｓ）、要素Ｖ５０［ｃ２］をー１に設定する（ステップＳ６１５）。

【０２５９】これらの要素Ｖ５０［ｃ２］が０であるかの判定処理（ステップＳ６１４）、ｃ２←（ｃ２＋１）ｍｏｄ（ｎ）の処理（ステップＳ６１３）、およびＶ５０［ｃ２］←－１の処理（ステップＳ６１５）により、Ｖ５０［ｃ２］の要素が０でない場合に、Ｖ５０［ｃ２］の右側の要素で０であるものを巡回移動しながら順に探して－１に設定する。

【０２６０】そして次に、配列出力部４００は、カウンタｃ１’のカウント値ｃ１がｃ１＜ｎ２－１であるかの判定を行う（ステップＳ６１６）。

【０２６１】配列出力部４００は、ｃ１＜ｎ２－１であると判定したとき（ステップＳ６１６のＹｅｓ）、カウント値ｃ１をカウントアップすると共に（ｃ１←ｃ１＋１）、カウント値ｃ２をｃ２←（ｃ２＋Ｄ［ｃ１＋ｎ１］）ｍｏｄ（ｎ）とし（ステップＳ６１７）、そして再度、Ｖ５０［ｃ２］が０であるかの判定を行う（ステップＳ６１４）。

【０２６２】一方、配列出力部４００は、ｃ１＜ｎ２－１でないと判定したとき（ステップＳ６１６のＮｏ）、配列Ｖ５０の－１の要素がｎ２個になったとして配列Ｖ５０を出力する。

【０２６３】次に、図１５のフローチャートにそって具体例を説明する。図１７は配列出力部４００における配

列Ｖ５０の各段階での配列状態を示している。

【０２６４】配列Ｖ５０は２５１次元の配列（ｎ＝２５１）であり、ｎ１＝５０、ｎ２＝５０とする。また、整数Ｘを８ビットごとに分割したときの分割情報を例えば、分割情報Ｄ［０］＝１３９、分割情報Ｄ［１］＝１３０とする。

【０２６５】図１５のフローチャートによる流れで、本具体例に従えば、カウント値ｃ２＝Ｄ［０］＋１＝１４０となる（ステップＳ６０５）。

【０２６６】そして、配列Ｖ５０は全要素が０であるため、配列Ｖ５０の左から１４０番目の要素Ｖ５０［１４０］を１に設定する（ステップＳ６０８）。この配列状態は、図１７に示す配列Ｖ５１の通りであり、左から１４０番目の要素が１であり、その他の要素が０となっている。

【０２６７】次に、配列Ｖ５０の１の要素は５０個でないため（ステップＳ６０９のＮｏ）、配列Ｖ５０の要素を１に設定する処理をさらに行うため、カウント値ｃ２をｃ２←（ｃ２＋Ｄ［１］）ｍｏｄ（ｎ）とする（ステップＳ６１０）。カウント値ｃ２＝（１４０＋１３０）ｍｏｄ２５１＝１９ｍｏｄ２５１となり、配列Ｖ５１の左から１９番の要素Ｖ５１［１９］を１にする。この配列状態は、図１７に示す配列Ｖ５２の通りであり、左から１９番目と１４０番目の要素が１であり、その他の要素が０となっている。

【０２６８】このように、配列Ｖ５０の各要素に１を設定する処理をするとき、設定する位置の要素、例えばＶ５０［１２０］がすでに１となっている場合は、その右側の要素を巡回移動して０の要素を探し、一番初めの０の要素を１に設定する。そして、Ｖ５０［２５１］までのすべての要素が０でない場合は、巡回移動して一番左のＶ５０［１］の要素に戻り、０の要素を１に設定する。同様にして、配列Ｖ５０の－１の要素を順に設定する。

【０２６９】上述したように、配列出力部４００は、配列Ｖ５０における最初の１の要素を整数Ｘより求まる分割情報Ｄ［０］に基づいて一様に決定する。そして、最初に決定された１の要素の場所から整数Ｘにより求まる分割情報Ｄ［ｉ］を基に順に次の要素を設定する位置を決定して、ｌあるいは－１を設定するため、配列Ｖ５０は、整数Ｘから一様に分布する。

【０２７０】このとき、配列出力部４００は、配列Ｖ５０におけるｎ１個の１の要素と、ｎ２個の－１の要素を決定するために、入力される整数Ｘは、８ビットの分割情報を（ｎ１＋ｎ２）個必要である。そのために、整数Ｘは、配列Ｖ５００各要素を十分設定できる大きいものを設計段階で選択すればよい。

【０２７１】このように、本実施の形態によれば、配列出力部４００は、整数Ｘに基づいて、一様にｎ次元の配列を出力することができる。そのため、ＮＴＲＵ暗号に

ＦＯＳＲＴを適用した場合において、図１（ａ）の暗号化装置１０の配列出力部１００の代わりにこの配列出力部４００とし、ハッシュ関数部４０より出力されるハッシュ関数値Ｈ（ｍ）に基づいて、配列出力部４００が一様にｎ次元の配列を出力することで、ハッシュ関数の分布の一様性を保持することが可能となり、暗号化装置１０で生成する暗号文の安全性を高めることができる。

【０２７２】さらに、配列出力部４００は、整数Ｘのみの情報より配列Ｖ５０を設定しているため、メモリテーブルを用いる必要がなく、メモリが少なくてよいという効果が得られる。

【０２７３】また、図１（ｂ）の復号化装置１５の配列出力部１０５の代わりに、この配列出力部２００を用いることで暗号化された暗号文の復号が可能となる。

【０２７４】尚、配列出力部１００を暗号化装置１０に用いて、ハッシュ関数値を基に配列を出力する構成としたがこれに限定されない。

【０２７５】また、各実施の形態で説明した暗号化装置１０は、図１８に示す携帯電話機５００内に搭載されて使用されたり、また、インターネット上の電子決済、電子商取引として使用されたりする。

【０２７６】また、各実施の形態１、２、３、４において、各配列出力部は、ｎ１個の要素が１、ｎ２個の要素が－１、その他の要素が０の配列を出力しているが、１、－１が他の数であってもよい。また、各実施の形態１、２、３、４において、各配列出力部は、１、－１、０の３値の配列を出力するが、２値であってもよいし、４値以上であってもよい。

【０２７７】また各実施の形態１、２、３、４のうちのいずれかを使用する暗号方法であってもよい。

【０２７８】

【発明の効果】以上の説明から明らかなように、本発明に係る配列出力装置においては、入力された整数に依存して、Ｋ個の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力装置であって、初期配列を暫定的に決定する初期配列決定手段と、前記入力された整数に基づいて、前記初期配列決定手段が決定した前記初期配列の配列要素を改編する改編手段とを備えることとを特徴とする。

【０２７９】これによって、多くのメモリを使用することなく、整数値に基づいて一様にｎ次元の配列を得ることができ、ハッシュ関数値などの一様に分布された整数値に基づいてその一様性を保持してｎ次元の配列を得ることが可能となる。

【０２８０】また、前記改編手段は、前記入力された整数を所定の整数で除算し剰余を求める除算部と、前記除算部が求めた剰余に基づいて、前記初期配列の配列要素を置換する置換部とを備えるようにしてもよい。

【０２８１】これにより、ハッシュ関数値などの一様に分布された整数値に基づいてその一様性を保持してｎ次

元の配列を得ることが可能となる。

【０２８２】また、前記改編手段は、前記入力された整数を所定の整数で除算し剰余を求める除算部と、前記初期配列における整数Ｐ３の配列要素のうち前記除算部が求めた剰余に基づく位置の配列要素を整数Ｐ１に置き換える整数配置部とを備えるようにしてもよい。

【０２８３】これにより、ハッシュ関数値などの一様に分布された整数値に基づいてその一様性を保持してｎ次元の配列を得ることが可能となる。

【０２８４】また、メッセージを暗号化する暗号化装置であって、メッセージを一方向関数で演算し、その結果を関数値として出力する関数値出力手段と、初期配列を暫定的に決定する初期配列決定手段、および前記関数値出力手段が出力する関数値に基づいて、前記初期配列決定手段が決定した前記初期配列の要素を改編する改編手段を備え、前記関数値に依存して、Ｋ値の整数の組み合わせからなるｎ次元の様々な配列を出力する配列出力手段と、前記配列出力手段が出力する配列を基に、暗号文を生成する暗号文生成手段とを備えることを特徴とする。

【０２８５】これにより、メッセージ値のハッシュ関数値などの、一方向関数により一様に分布された整数値に基づいて、その一様性を保持してｎ次元の配列を得ることが可能となり、暗号文の安全性を高くすることができる。

【図面の簡単な説明】

【図１】本発明における第１の実施の形態に係る（ａ）は暗号化装置の構成を示すブロック図であり、（ｂ）は復号化装置の構成を示すブロック図である。

【図２】同上の配列出力部の構成を示すブロック図である。

【図３】同上の配列出力部から出力される配列の配列状態を示す図である。

【図４】同上の配列出力部の動作を示すフローチャートである。

【図５】本発明における第２の実施の形態に係る配列出力部の構成を示すブロック図である。

【図６】同上の配列出力部における第１の数配置部の動作を示すフローチャートである。

【図７】同上の配列出力部における第１の数配置部から出力される配列の配列状態を示す図である。

【図８】同上の配列出力部における第２の数配置部の動作を示すフローチャートである。

【図９】同上の配列出力部における第２の数配置部から出力される配列の配列状態を示す図である。

【図１０】本発明における第３の実施の形態に係る配列出力部の構成を示すブロック図である。

【図１１】同上の配列出力部における第１の数配置部から出力される配列の配列状態を示す図である。

【図１２】同上の配列出力部における第２の数配置部から出力される配列の配列状態を示す図である。

【図１３】（ａ）（ｂ）（ｃ）はいずれも同上の配列出力部の各部で設定される配列の配列状態を示す図である。

【図１４】本発明における第４の実施の形態に係る配列出力部の構成を示す図である。

【図１５】同上の配列出力部の動作を示すフローチャートである。

【図１６】同上の配列出力部に入力される整数を示す図である。

【図１７】同上の配列出力部から出力される配列の配列状態を示す図である。

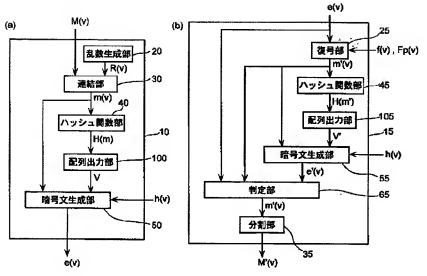
【図１８】本発明の配列出力部を有する暗号化装置が搭載される携帯電話機の外観を示す外観図である。

【図１９】従来の配列出力方法を示すフローチャートである。

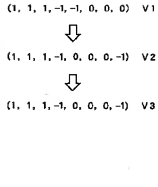
【符号の説明】

１０　暗号化装置
１５　復号化装置
２０　乱数生成部
２５　復号部
３０　連結部
３５　分割部
４０　ハッシュ関数部
４５　ハッシュ関数部
５０　暗号文生成部
５５　暗号文生成部
６５　判定部
１００　配列出力部
１１０　初期配列決定部
１２０　配列要素置換部
２００　配列出力部
２１０　第１の数配置部
２２０　第２の数配置部
３００　配列出力部
３１０　第１の数配置部
３２０　第２の数配置部
４００　配列出力部
５００　携帯電話機

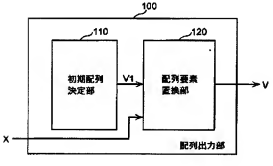
【図１】



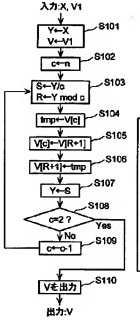
【図３】



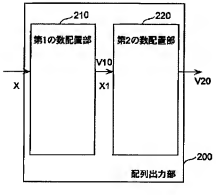
【図２】



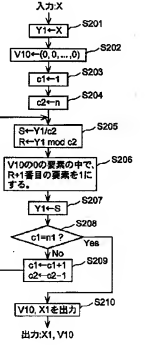
【図４】



【図５】

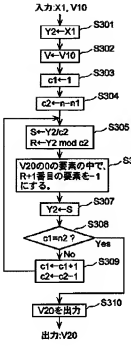


【図６】

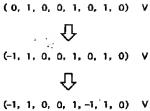


【図７】

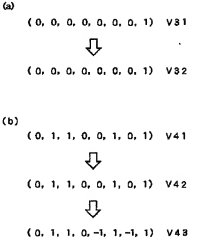
【図８】



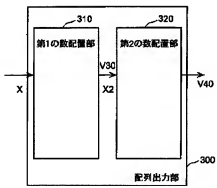
【図９】



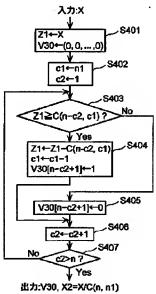
【図１３】



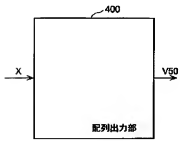
【図１０】



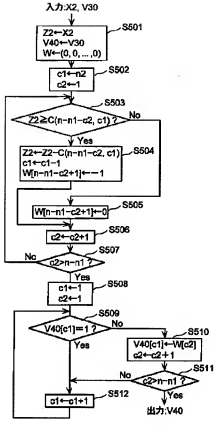
【図１１】



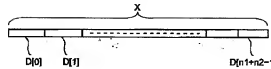
【図１４】



【図１２】



【図１６】

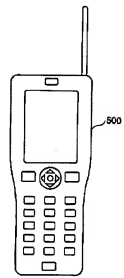


【図１７】

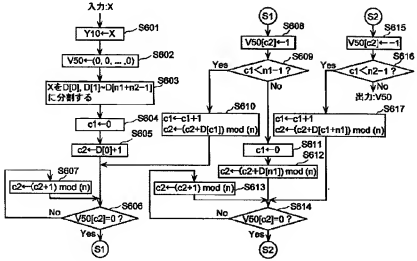
(0, 0, 0, 0, ・・・・0, 1, 0, ・・・・0) V51

(0, 0, 1, 0, ・・・・0, 1, 0, ・・・・0) V52

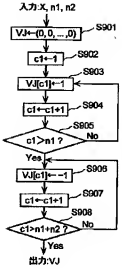
【図１８】



【図１５】



【図１９】



フロントページの続き

(72)発明者　横田　薫
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　館林　誠
大阪府門真市大字門真1006番地　松下電器産業株式会社内

Ｆターム(参考)　5J104 AA18 JA21 NA12 NA39